

Dynamics of Large Networks

Jurij Leskovec

September 2008
CMU-ML-08-111



Dynamics of Large Networks

Jurij Leskovec

September 2008

CMU-ML-08-111

Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:

Christos Faloutsos, Chair

Avrim Blum

John Lafferty

Jon Kleinberg, Cornell University

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Copyright © 2008 Jurij Leskovec

This work was partially supported by the National Science Foundation under Grants No. SENSOR-0329549, IIS-0534205, and IIS-0705359, by Microsoft Research Graduate Fellowship, and the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contracts No.W-7405-ENG-48 and DE-AC52-07NA27344 (LLNL-CONF-404625). This work was also partially supported by the Pennsylvania Infrastructure Technology Alliance (PITA), a Yahoo Research Alliance Gift, a SPRINT gift, with additional funding from Intel, NTT and Hewlett-Packard.

Keywords: network analysis, graph mining, complex networks, network evolution, small-world phenomena, densification, Kronecker graphs, information propagation, diffusion, cascades, viral marketing, outbreak detection, graph partitioning, network community structure

*Samo kdor vidi svoje korenine,
premeri debla moč in vej višine,
ve, kako visoko je nebo.*

Abstract

A basic premise behind the study of large networks is that interaction leads to complex collective behavior. In our work we found very interesting and counterintuitive *patterns* for *time evolving networks*, which change some of the basic assumptions that were made in the past. We then develop *models* that explain processes which govern the network evolution, fit such models to real networks, and use them to generate realistic graphs or give formal explanations about their properties. In addition, our work has a wide range of applications: it can help us spot anomalous graphs and outliers, forecast future graph structure and run simulations of network evolution.

Another important aspect of our research is the study of “local” patterns and structures of propagation in networks. We aim to identify building blocks of the networks and find the patterns of influence that these blocks have on information or virus propagation over the network. Our recent work included the study of the spread of influence in a large person-to-person product recommendation network and its effect on purchases. We also model the *propagation of information* on the blogosphere, and propose *algorithms* to efficiently find influential nodes in the network.

A central topic of our thesis is also the analysis of *large datasets* as certain network properties only emerge and thus become visible when dealing with lots of data. We analyze the world’s largest social and communication network of Microsoft Instant Messenger with 240 million people and 255 billion conversations. We also made interesting and counterintuitive observations about network community structure that suggest that only small network clusters exist, and that they merge and vanish as they grow.

Acknowledgments

First, I would like to thank my advisor Christos Faloutsos for his advice and support. This thesis would not have been possible without him and without freedom and encouragement he has given me over the last four years and the summer I spent at Carnegie Mellon.

I have had an amazing group of coauthors and collaborators. Each of them deserves my gratitude: Lada Adamic, Lars Backstrom, Deepay Chakrabarti, Anirban Dasgupta, Susan Dumais, Natalie Glance, Carlos Guestrin, Eric Horvitz, Bernardo Huberman, Matt Hurst, Jon Kleinberg, Andreas Krause, Ravi Kumar, Kevin Lang, Michael Mahoney, Mary McGlohon, Purna Sarkar, Ajit Singh, Andrew Tomkins and Jeanne VanBriesen. I would especially like to thank Jon for the collaboration during my first year at CMU and everything that followed; for great ideas, insights that make you smile, and advice about research and academic life. I learned a lot from Andreas and Carlos. Working with them was a great experience that brought together civil engineering and computer science, water and blogs. Andreas has been a great collaborator, coauthor and supportive friend. Thank you for the unselfish help, insights, feedback, and for making the job search a collaborative effort.

I have spent great summers at HP Labs, Microsoft Research and Yahoo! Research. During this time basically half of the chapters of this thesis got done. Working with “physicists” Lada and Bernardo taught me to think differently about the problems and ask different questions. Thank you also for the “big machine” that made me master and appreciate intimidatingly large datasets. I would also like to thank Eric and Sue for the immense enthusiasm and support, and for the trust in studying the world’s social network. Also, thank you for the carrier advice, and the help with making a “perfect” research statement. It really worked! I also thank Michael, Ravi and Andrew for fun problems with many implications.

I would like to thank my thesis committee members, Avrim Blum, Jon Kleinberg and John Lafferty for their advice and comments.

Of course, life would not be the same without the fellow machine learners, housemates and friends Andrew Arnold, Thomas Latoza and Brian Ziebart. I will never forget long nights when we were doing homework or just playing Setters. I also acknowledge CMU machine learners and Machine Learning Department for making it such a stimulating and interactive environment. Thanks goes also to Diane Stidle and Charlotte Yano for patiently answering all my emails, requests and endless travel reimbursements.

I would also like to thank the Slovenian gang: Marko Grobelnik, Dunja Mladenčič, Blaž Fortuna, Blaž Novak and Janez Brank from Jožef Stefan Institute. I am especially grateful to Marko for his tremendous support, enthusiasm and good will. For guiding me through my first research steps, for advice and long discussions about life, universe and everything. Meeting him 12 years ago while I was “bored” in high-school really set the course for what is here today.

Finally, I thank my parents and sisters, Nina and Darja, for endless love, encouragement, advice and support. Brez vas se ne bi vračal domov tako rad, kot se. Hvala tudi vsem Silakom in Šentjoštu za bogatost naših življenj. Brez vas vseh to delo in prehojena pot ne bila to, kar sta. Hvala vsem!

Contents

1	Introduction	1
1.1	Motivation and applications	2
1.1.1	Network evolution	4
1.1.2	Network cascades	4
1.1.3	Large data	5
1.2	Thesis overview and contributions	6
1.2.1	Part 1 – Network evolution	6
1.2.2	Part 2 – Network cascades	8
1.2.3	Part 3 – Large data	9
2	Overview and survey	11
2.1	Basic concepts and definitions	11
2.1.1	General graph-theoretic concepts	11
2.1.2	Diameter and effective diameter	12
2.1.3	Power law distributions and heavy tails	13
2.2	Statistical properties of networks	16
2.3	Models of network structure and evolution	19
2.4	Diffusion and cascading behavior in networks	22
2.4.1	Information cascades in blogosphere	23
2.4.2	Cascades in viral marketing	23
2.5	Table of symbols	24
2.6	Table of datasets	24
I	Network evolution	25
3	Macroscopic network evolution	27
3.1	Introduction	27
3.2	Related work on graphs over time	30
3.3	Observations	31
3.3.1	Densification Laws	32
3.3.2	Shrinking Diameters	39
3.3.3	Does densification cause shrinking diameter?	43
3.3.4	Densification and the degree distribution over time	44
3.4	Proposed models	50
3.4.1	Community Guided Attachment	50

3.4.2	The Forest Fire Model	55
3.5	Discussion	63
4	Microscopic network evolution	65
4.1	Introduction	65
4.1.1	Evaluation based on likelihood	66
4.1.2	Data and model structure	66
4.1.3	Our results	67
4.2	Relation to previous work on network evolution	68
4.3	Preliminaries	68
4.3.1	Datasets	68
4.3.2	Notation	69
4.3.3	Maximum-likelihood principle	69
4.4	Preferential attachment	70
4.4.1	Edge attachment by degree	70
4.4.2	Edges by the age of the node	72
4.4.3	Bias towards node age and degree	73
4.5	Locality of edge attachment	74
4.5.1	Triangle-closing models	77
4.6	Node and edge arrival process	79
4.6.1	Edge initiation	79
4.6.2	Node arrivals	83
4.7	A network evolution model	84
4.7.1	Gaps and power law degree distribution	85
4.7.2	Validation of the model	86
4.7.3	Unfolding network evolution	87
4.8	Discussion	88
5	Kronecker graphs	89
5.1	Introduction	89
5.2	Relation to previous work on network modeling	91
5.2.1	Graph Patterns	92
5.2.2	Generative models of network structure	93
5.2.3	Parameter estimation of network models	93
5.3	Kronecker graphs model	94
5.3.1	Main idea	94
5.3.2	Analysis of Kronecker Graphs	98
5.3.3	Stochastic Kronecker Graphs	101
5.3.4	Additional properties of Kronecker graphs	103
5.3.5	Two interpretations of Kronecker graphs	103
5.3.6	Fast generation of Stochastic Kronecker Graphs	106
5.3.7	Observations and connections	106
5.4	Simulations of Kronecker graphs	107
5.4.1	Comparison to real graphs	107
5.4.2	Parameter space of Kronecker Graphs	109
5.5	Kronecker graphs model estimation	110
5.5.1	Preliminaries	111

5.5.2	Problem formulation	112
5.5.3	Summing over the node labelings	114
5.5.4	Efficiently evaluating likelihood and gradient	117
5.5.5	Calculating the gradient	118
5.5.6	Determining the size of initiator matrix	118
5.6	Experiments on real and synthetic data	119
5.6.1	Permutation sampling	119
5.6.2	Properties of the optimization space	124
5.6.3	Convergence of the graph properties	124
5.6.4	Fitting to real-world networks	126
5.6.5	Fitting to other large real-world networks	131
5.6.6	Scalability	133
5.7	Discussion	134
5.8	Conclusion	136

II Network cascades 139

6	Diffusion and cascading behavior in viral marketing	141
6.1	Introduction	141
6.2	Connection to viral marketing	143
6.3	The recommendation network	145
6.3.1	Recommendation program and dataset description	145
6.3.2	Identifying successful recommendations	147
6.3.3	Properties of the recommendation network	147
6.3.4	Recommendation network over time	149
6.3.5	Preliminary observations and discussion	151
6.4	Propagation of recommendations	152
6.4.1	Forward recommendations	152
6.4.2	Identifying cascades	154
6.4.3	The recommendation propagation model	156
6.5	Success of Recommendations	157
6.5.1	Human adoption curve	157
6.5.2	Success of subsequent recommendations	159
6.5.3	Success of outgoing recommendations	160
6.5.4	Success of incoming recommendations	161
6.6	Timing of recommendations and purchases	164
6.7	Recommendations and communities of interest	165
6.7.1	Communities and purchases	165
6.7.2	Recommendation effectiveness by book category	168
6.8	Products and recommendations	170
6.8.1	How long is the long tail?	170
6.8.2	Modeling the product recommendation success	171
6.9	Cascade shapes in viral marketing	174
6.9.1	Cascades	174
6.9.2	The recommendation network	175
6.9.3	Proposed method	176

6.9.4	Patterns of recommendation	178
6.10	Conclusion	182
7	Information propagation on the blogosphere	184
7.1	Introduction	184
7.1.1	Summary of findings and contributions	185
7.1.2	Chapter organization	185
7.2	Connection to temporal modeling and epidemiology	185
7.2.1	Burstiness and power laws	185
7.2.2	Blogs	186
7.2.3	Information cascades and epidemiology	186
7.3	Preliminaries	187
7.4	Experimental setup	189
7.4.1	Dataset description	189
7.4.2	Data preparation and cleaning	189
7.5	Observations, patterns and laws	190
7.5.1	Temporal dynamics of posts and links	190
7.5.2	Blog network topology	193
7.5.3	Post network topology	194
7.5.4	Patterns in the cascades	194
7.6	Proposed model and insights	198
7.6.1	Cascade generation model	198
7.6.2	Validation of the model	199
7.6.3	Variations of the model	200
7.7	Discussion	201
7.8	Conclusion	201
8	Outbreak and cascade detection	203
8.1	Introduction	203
8.2	Outbreak Detection	205
8.2.1	Problem statement	205
8.2.2	Placement objectives	207
8.2.3	Properties of the placement objectives	208
8.2.4	Multicriterion optimization	208
8.3	Proposed algorithm	209
8.3.1	Bounds for the algorithm	209
8.3.2	Online bounds for any algorithm	210
8.4	Scaling up the algorithm	211
8.4.1	Speeding up function evaluations	211
8.4.2	Reducing function evaluations	212
8.5	Case study: Blog Network	213
8.5.1	Experimental setup	214
8.5.2	Objective functions	214
8.5.3	Solution quality	215
8.5.4	Cost of a blog	215
8.5.5	Comparison to heuristic blog selection	217
8.5.6	Fractionally selecting blogs	218

8.5.7	Generalization to future data	218
8.5.8	Scalability	219
8.6	Case study: Water networks	220
8.6.1	Experimental setup	220
8.6.2	Objective functions	220
8.6.3	Solution quality	220
8.6.4	Multicriterion optimization	221
8.6.5	Scalability	223
8.7	Discussion and connection to previous work	224
8.7.1	Relationship to Influence Maximization	224
8.7.2	Optimizing submodular functions	224
8.7.3	Virus propagation and outbreak detection	225
8.7.4	Information cascades and blog networks.	225
8.7.5	Water distribution network monitoring.	225
8.8	Conclusions	226
III	Large data	228
9	MSN Messenger communication network	230
9.1	Introduction	230
9.2	Instant Messaging	231
9.2.1	Research on Instant Messaging	232
9.2.2	Data description	232
9.3	Usage & population statistics	233
9.3.1	Levels of activity	233
9.3.2	Demographic characteristics of the users	235
9.4	Communication characteristics	237
9.5	Communication demographics	238
9.5.1	Communication by age	238
9.5.2	Communication by gender	239
9.5.3	World geography and communication	241
9.5.4	Communication among countries	243
9.5.5	Communication and geographical distance	244
9.6	Homophily of communication	248
9.7	The communication network	249
9.7.1	How small is the small world?	252
9.7.2	Network cores	253
9.7.3	Strength of the ties	254
9.8	Conclusion	256
10	Network community structure	258
10.1	Introduction	258
10.1.1	Overview of our approach	259
10.1.2	Summary of our results	262
10.1.3	Outline of the chapter	266
10.2	Background on communities and overview of our methods	266

10.2.1	Social and information network datasets we analyze	267
10.2.2	Clusters and communities in networks	269
10.2.3	Approximation algorithms for finding low-conductance cuts	271
10.3	The Network Community Profile Plot (NCP plot)	272
10.3.1	Definitions for the network community profile plot	272
10.3.2	NCP plots for small social networks, expander and low-dimensional graphs	273
10.3.3	NCP plots for large social and information networks	277
10.3.4	More community profile plots for large social and information networks	280
10.4	More structural observations of our network datasets	280
10.4.1	General statistics on our network datasets	284
10.4.2	Network “whiskers” and the “core”	284
10.4.3	Bags of whiskers and communities of composed whiskers	285
10.4.4	NCP of networks with no 1-whiskers	288
10.5	Comparison to other algorithms	290
10.5.1	Cross-checking between algorithms	290
10.5.2	Lower bounds on cut conductance	293
10.5.3	Local Spectral and Metis+MQI	294
10.6	Models for network community structure	297
10.6.1	NCP plots for commonly-used network generation models	298
10.6.2	Very sparse random graphs have very unbalanced deep cuts	300
10.6.3	An intuitive toy model for generating an upward-sloping NCP plot	306
10.6.4	A more realistic model of network community structure	307
10.7	Discussion	311
10.7.1	Comparison with “ground truth” and sociological communities	311
10.7.2	Connections and broader implications	314
10.7.3	Relationship with community identification methods	316
10.7.4	Relationship with other theoretical work	317
10.8	Conclusion	318
11	Web projections: Learning from contextual subgraphs of the web	320
11.1	Introduction	320
11.2	Query projections	322
11.2.1	Query projection and connection graphs	323
11.2.2	From graph to evidential features	323
11.3	Generating case libraries	324
11.3.1	Web as a graph	326
11.3.2	Human-rated search results	327
11.3.3	Query reformulation corpus	327
11.4	Quality of search results	328
11.4.1	Problem definition	328
11.4.2	Experimental setup	328
11.4.3	Relative quality of result sets	329
11.4.4	Absolute quality of a result set	331
11.5	Query reformulations	333
11.5.1	Experimental setup	333
11.5.2	Probability of query reformulation	333
11.5.3	Query specialization versus generalization	334

11.5.4 Predicting type of query reformulation	337
11.6 Connections to prior research	338
11.7 Summary and directions	339
IV Conclusion and future directions	342
12 Conclusion	343
12.1 Summary of contributions	344
12.2 Vision for the future	345
12.2.1 Medium term goals	346
12.2.2 Long-term goals	347
A Appendix	348
A.1 Table of symbols	348
A.2 Datasets and basic statistics	350
Bibliography	354

Chapter 1

Introduction

The main interest of our research has been in understanding the structural properties and patterns in the evolution of large graphs and networks. What does a “normal” network look like? How will it evolve over time? How can we spot “abnormal” interactions (*e.g.*, spam) in a time-evolving e-mail graph? How do information and viruses spread over the network? How can we identify and find influential nodes or select nodes to immunize in networks? Answers to such questions are vital to a range of application areas from the identification of illegal money-laundering rings, misconfigured routers on the Internet, viral marketing and protein-protein interactions to disease outbreak detection.

A basic premise behind the study of large networks is that interaction leads to complex collective behavior. We study three such cases where complex collective behavior emerges from local interaction:

- **Network evolution:** The study of statistical properties and models that govern the generation and evolution of large real-world networks. Evolution of network structure is a form of collective behavior, where our studies are the first to examine network evolution over long time periods both at the macroscopic level of statistical network properties and at the microscopic level by analyzing individual arrivals and attachments of millions of edges. We view the network as a big complex system, and observe its static and temporal properties and patterns to design models that capture and help us understand the temporal and static patterns of real-world networks.
- **Network cascades:** The study of the network by starting from individual nodes and small communities. Cascades are a form of collective behavior that has been analyzed both empirically and theoretically, but for which the study of complete, large-scale datasets has been limited. We examine two examples where it is possible to directly observe and measure large scale cascading behavior. We show that cascades exist in large real-world networks, and investigate some of their structural features. We aim to find common and abnormal sub-network patterns and understand the propagation of influence, information, diseases and computer viruses over the network. Once we know the propagation patterns and structure, we devise algorithms for efficiently finding influential nodes and detecting disease or virus outbreaks in networks.
- **Large data:** The study of large real-world networks with hundreds of millions of nodes and edges. Working with such datasets is important in order to understand and take into account performance and scalability issues and to discover patterns that may become apparent only in massive datasets. For example, we demonstrate the value of large data in the case of quantifying network community structure where most of the existing work focused on small networks of several hundred nodes. On

the other hand we analyze large networks of millions of nodes and show their structure is fundamentally different from small networks. Basically, our observations only become possible when working with enough data so that the behavior or the structural property emerges.

1.1 Motivation and applications

Traditionally small networks were analyzed from a “node centric” point of view where researchers wanted to answer questions about behavior and properties of particular nodes in the network. Though such models are very expressive, they often fail to scale to large networks with millions of nodes and edges. Moreover, many times we need to work with a large network for a structural property of the network to emerge; thus, the focus moves to the study of structural properties of the network as a whole.

Today with the ubiquity of the web and with billions of its users there are several opportunities to study phenomena and computing systems at scales that were not possible before. This can be summarized by the following three points:

- On-line computing systems (*e.g.*, web, email) have detailed traces of human activity.
- Such applications (*e.g.*, Facebook, Second Life, blogs) have millions of users interacting with one another and with the system.
- Such rich data can naturally be modeled and represented as a network.

For example, Web 2.0 [O'Reilly, 2005] is a set of tools that enable the masses to easily create content on the WWW, in the form of blogs, social networks, video and photo collections, and simple application creation frameworks. In addition, Web 2.0 has amplified the importance of relationships between users that are represented in social networks. The emergence of this new and varied content has led to a flurry of research activity that aims to mine the content and infer useful data from it (*e.g.*, sentiment analysis, network analysis). Other such examples include: mobile caller networks, which include traces of calling and mobility dynamics of millions of people; Instant messaging data that in a single application under a single system captures communication patterns of basically the whole planet Earth; Or for example, online worlds and massively multiplayer online games which are capable of supporting hundreds or thousands of players interacting simultaneously.

This presents many unique opportunities and challenges. On one hand, it presents a shift in computer science from engineering big systems to a more natural science approach. Unlike other areas in the field, we are not engineering a system over which we have complete control anymore. We are studying the real world, adding local mechanisms to achieve certain global goals. On the other hand, the emergence of socially rich computing applications with millions of users allows us to ask questions that were impossible to answer before as large scale human social dynamics data was practically impossible to collect. Moreover, this also offers a unique opportunity for computer science to reach towards other sciences like social sciences, economics and mechanism design, and physics of complex systems.

To understand the complex behavior and dynamics of the web or the internet backbone one basically follows the steps of the scientific method. Thus, throughout this thesis we follow the following three steps:

Thesis part	Steps of the thesis		
	1: Observations	2: Models	3: Algorithms
Part 1: Network evolution	chapter 3	chapter 4	chapter 5
Part 2: Network cascades	chapter 6	chapter 7	chapter 8
Part 3: Large data	chapter 9	chapter 10	chapter 11

Table 1.1: Structure of the thesis with references to the chapters.

- **STEP 1 – Observations:** *Hypothesis and data analysis.* We consider a problem of interest and form a hypothesis. We collect real data, measure and observe the phenomena of interest, and perform measurements and analyses that prove or disprove the hypothesis.
- **STEP 2 – Models:** *Explanation/model design.* Given a novel observation we design models that give intuitions, explanations and predictions about the system.
- **STEP 3 – Algorithms:** *System and algorithm development.* Using insights from the data analysis and models that explain the observations, we develop new better and faster algorithms and systems.

While the first two steps are part of usual scientific method, the last third step is somewhat unique to computer science as it introduces a feedback loop to the process. It aims to harness the empirical observations and intuitions coming from the models to develop better systems, applications and algorithms. This is also the primary reason why computer science *per se* is interested in asking and validating the empirical questions.

For example, recently the field of “internet measurement” [Crovella and Bestavros, 1997, Zhang et al., 2002] emerged in the area of computer networks. It empirically explores, measures and models how Internet as a whole looks, works, and behaves. This is shift from traditional engineering point of view. If one engineers a system, there is usually no need to measure and model it as we designed it and thus understand how it works. However, even though the physical Internet was designed and engineered, it evolved into a large and complex system that one today needs to measure and model to understand it and make predictions about it. Thus also comparisons and parallels of the internet with complex physical and biological systems.

This puts computer science a unique position as we not only study but also design and build such complex systems. We are not only silent observers that measure and model, but we can also design, create and impose rules and incentives on such systems. Via computing application we have control at micro level, while the system is affected globally. So it is important to understand how such systems work, and understand what consequences our micro decisions have globally, which then naturally closes the loop between design and engineering on one hand, and empirical measurement and modeling on the other hand.

Thus, the thesis naturally breaks into nine pieces, as shown in Table 1.1: the rows correspond to the research problems, and the columns correspond to the steps of the scientific process as described above. Next we give the motivation for each of the nine parts, following by the summary of our contributions.

1.1.1 Network evolution

Ultimately we search for interesting measures that let us characterize the network structure and the processes spreading over the networks. Then we design models and algorithms that take advantage of the identified structural network properties.

The focus of analyzing and modeling the structure of large networks aims to do the following three things:

- (1) **Observations:** *What are interesting statistical properties of network structure?* The aim is to find statistical properties, such as path lengths and degree distributions, that characterize the structure and behavior of networks, and suggest appropriate ways to measure these properties.
- (2) **Models:** *What is a good model that helps us understand these properties?* We aim to create models of networks that can help us to understand the meaning of the statistical properties of networks. How they come to be as they are, and how they interact with one another?
- (3) **Algorithms:** *Estimate the model and predict behavior of networks based on measured structural properties and local rules governing individual nodes?* How, for example, will Internet structure evolve and how does the network structure affect traffic on the Internet or performance of a web crawler?

Applications:

- *Models and parameters:* Generative models and their parameters give us insight into the graph formation process. Intuitions developed by the models are useful in understanding the network generation processes and reasoning about the structure of the networks in general.
- *Graph generation:* Our methods form means of assessing the quality of graph generators. Synthetic graphs are important for “what if” scenarios where we need to extrapolate and simulate graph growth and evolution, since real graphs may be impossible to collect and track (like, e.g., a very large friendship graph between people). Synthetic graphs can then be used for predicting future network evolution, hypothesis testing, and simulations and evaluation of algorithms, e.g., simulations of new network routing protocols, virus propagation, etc.
- *Extrapolations and predictions:* For several real graphs, we have a lot of snapshots of their past. What can we say about their future? Our results help form a basis for validating scenarios for graph evolution.

1.1.2 Network cascades

The second part of the thesis deals with information propagation in large networks. The social network of interactions among a group of individuals plays a fundamental role in the spread of information, ideas, and influence. Such effects have occurred in many cases, when an idea or action gains sudden widespread popularity through word-of-mouth or “viral marketing” effects. To take a recent example from the technology domain, free e-mail services such as Microsoft’s Hotmail and later Google’s Gmail achieved wide usage largely through referrals, rather than direct advertising. However, directly measuring such behaviors on a large scale proved difficult.

We would like to understand how the structure of the network affects the spread of information, influence and viruses over the network. We monitor the spread of information on the blogosphere or recommendations in a product recommendation network. We aim to answer the following questions:

- (1) **Observations:** *What are the typical patterns of information propagation?* The aim is to find statistical properties, such as how deep or wide are the propagation graphs (also called *cascades*) or how fast is the information spreading? We want to characterize such behaviors and suggest appropriate ways to measure them.
- (2) **Models:** *What is a good model that helps us understand these properties?* For example, we aim to create models of information propagation on the web. Why information or diseases spread in a particular way, and how does this interact with the network structure?
- (3) **Algorithms:** *How to identify influential nodes and detect disease outbreaks?* For example, given a fixed budget of attention, which blogs should we read to be most up to date on the news? Or similarly, in a big water distribution network, where shall we position the sensors to detect disease outbreaks as quickly as possible?

Applications:

- *Cascade formation:* Understanding cascade formation helps to explain the propagation of information and viruses over the network. This allows for more accurate models of virus propagation, which can be used in epidemiology for simulations.
- *Outbreak detection:* Our work on cascades also gives us the means to study, for example, which nodes to inoculate to prevent a virus from spreading through the network, or where to place sensors in a water distribution network to quickly detect disease outbreaks.

1.1.3 Large data

A basic premise behind the study of networks is that interaction leads to collective behavior. For such collective behavior to become “visible” and detectable by statistical and machine learning methods one needs to analyze large datasets. As it turns out, many network properties follow heavy-tailed distributions that have infinite variances, which makes estimation hard and requires lots of data.

- (1) **Observations:** *What novel observations can we make from large datasets?* Using large datasets we can more accurately measure and experiment at scales that were not possible before. This can then lead to observing novel patterns or answering questions that were previously practically impossible to answer due to lack of data and tools to analyze them.
- (2) **Models:** *What is a good model that explains the observation?* When existing models fail to give an explanation, novel observations give us opportunities to design new models.
- (3) **Algorithms:** *How to handle and analyze large datasets?* Working with large datasets presents several engineering, systems and implementation challenges. It forces us to develop scalable parallel and out-of-core algorithms and tools that scale to large datasets and allow for measurement and analysis.

Applications:

- *Data mining*: Scaling data mining algorithms to large data is important by itself as it will allow us to discover novel patterns not found in smaller datasets.
- *Abnormality detection and computer network management*: In many network settings, “normal” behavior will produce subgraphs that obey properties of network growth. To detect activity which produces structures that deviate significantly from the normal patterns one needs to efficiently process lots of data. As the detections are made, we can flag them as abnormalities; this can potentially help with the detection of, *e.g.*, fraud, spam, or distributed denial of service (DDoS) attacks.

1.2 Thesis overview and contributions

The thesis addresses a number of important questions regarding the properties and patterns of large evolving networks by revealing how local behavior and structure lead to large scale phenomena.

The dissertation focuses on dynamics of time evolving networks, and the dynamics of processes, like virus propagation, that take place in networks. Our thesis has a “3-by-3” structure: it focuses on three problem domains where each of them is examined from three different aspects, *i.e.*, there are three parts: Network evolution, Network cascades and Large data, where each of them is composed of three chapters: Observations, Models and Algorithms. Table 1.1 gives the overall structure of our research with the mapping to the chapters of this thesis.

The main questions this thesis asks and answers are the following. We break each of them in the three steps the thesis follows:

1.2.1 Part 1 – Network evolution: How do real-world networks evolve?

Accurate properties of network growth, information propagation, and the models supporting them, have several possible consequences. Patterns give us ways for understanding and building models, and models help us to reason, monitor and predict features of the network in the future.

Step 1 – Observations: How do network properties evolve over time? (Chapter 3)

Here we examine how the macroscopic network properties, like diameter and network densification, change over time as the network evolves. This work had influence on thinking about fundamental structural properties of networks varying over time. For example, to date, it was commonly believed that the average degree of graphs of natural phenomena remains constant as they grow over time. Moreover, it was also assumed that the distances in networks slowly (logarithmically) increase with the network size. We showed that in fact networks *densify over time* as the number of edges $E(t)$ at time t is increasing as $E(t) \propto N(t)^a$ with the number of nodes $N(t)$. The densification exponent a is non-trivial, $a \approx 1.2\text{--}1.6$ [Leskovec et al., 2005b]. Even more surprisingly, the diameter of the network *shrinks* as it grows. These findings are fundamentally different from what was believed and commonly assumed in the past. A natural question to ask then is why do we observe these regularities? What is the connection between densification and shrinking diameters? As the existing intuitions and models do not explain these types of behavior, we developed a “Forest Fire” generative model that creates graphs with these proper-

ties [Leskovec et al., 2007b]. We also showed that densification itself is not enough to observe shrinking diameters.

Step 2 – Models: *How can we model the network growth and evolution?* (Chapter 4)

We examine network evolution by studying individual edge arrivals and placements. It is the individual edges that collectively give rise to observed macroscopic network properties. We use the *maximum-likelihood* principle to quantify the bias of new edges towards the degree and age of nodes, and to objectively compare various models such as preferential attachment. In fact, our work is the first to directly quantify the amount of preferential attachment in large social networks. We show that most new edges span very short distances, typically *closing triangles*. Motivated by these observations, we develop a *complete* model of *network evolution*, incorporating node arrivals, edge initiation, and edge destination selection processes. While node arrivals are mostly network-specific, the edge initiation process can be captured by exponential node lifetimes and a “gap” model based on a power law with exponential cutoff. We arrive at an extremely simple yet surprisingly accurate description of the edge destination selection in real networks.

Step 3 – Algorithms: *How can we generate large synthetic realistic looking networks?* (Chapter 5)

Last, we examine a question of how one can generate realistic looking synthetic graphs. This competency is important as we often need good null-models for simulations, what-if scenarios and hypothesis testing. We developed a Kronecker graph model that is based on the tensor product of graph adjacency matrices. In contrast to previous models, Kronecker graphs capture greatest number of static and dynamic network properties [Leskovec et al., 2005a], while being mathematically tractable. Moreover, we developed a maximum likelihood approach for parameter estimation of Kronecker graphs [Leskovec and Faloutsos, 2007]. Naive approaches take super-exponential time, while we developed a *linear* time parameter estimation algorithm. Using approximation and sampling we efficiently search the space of $10^{1,000,000}$ states, and estimate the model parameters for networks with millions of nodes in a matter of hours.

Contributions:

- We discovered the network *densification* and *shrinking diameter* that influenced the thinking about fundamental structural properties of networks varying over time.
- We developed Kronecker graphs, which are a *mathematically tractable* model of network generation and evolution. Moreover, Kronecker graphs are the first model that is able to capture *several* temporal and static network properties at the same time.
- We developed KRONFIT, an algorithm for estimating parameters of a Kronecker graphs model. Naive parameter estimation takes $O(N!N^2)$ time, while our approach scales *linearly* $O(E)$, which allows us to fit large graphs with millions on nodes and edges.

Impact:

- The work on densification and shrinking diameters received the best research paper award at ACM KDD 2005 [Leskovec et al., 2005b].
- Kronecker graphs have been harnessed by the high performance computing community, *e.g.*, by Jeremy Kepner [Kepner, 2008] at MIT Lincoln Lab, and David Bader at Georgia Tech, and Mohammad Mahdian from Yahoo! Research [Mahdian and Xu, 2007].

1.2.2 Part 2 – Network cascades: How information spreads in networks?

To model the evolution of large networked systems one also needs to understand how influence and information spread and propagate. Developing insights into such propagations is important for selecting targets for advertising and marketing, finding opinion makers with great influence in shaping people’s opinions, and to select nodes to monitor to best detect the potential epidemics.

The second part of the thesis presents our results on dynamics of processes that cascade from node to node like an epidemic. As the processes propagate they create *cascades* that are a form of collective behavior that has been analyzed both empirically and theoretically, but for which the study of complete, large-scale datasets has been limited. We investigated two examples of cascading behavior in networks where propagations naturally form cascades and we were able to directly measure and observe them on a large scale. In our work on information propagation between blogs [Leskovec et al., 2007d] and on product recommendation networks [Leskovec et al., 2006a, 2007a], we developed macroscopic models of the spread of influence in networks [Leskovec et al., 2007d], and found common and abnormal network substructures, called *cascades*, that the propagation process creates [Leskovec et al., 2007d, 2006b].

Step 1 – Observations: *What are patterns of diffusion and cascades in networks?* (Chapter 6)

First we present a study of influence and recommendation propagation in a large viral marketing network. To the best of our knowledge, our research was the first to answer a simple question: What is the probability of a person adopting the behavior (*e.g.*, buying a product) as more friends have adopted [Leskovec et al., 2006a]. Two competing theories are diminishing returns, which assumes that the probability of adoption increases slowly, and a critical threshold hypothesis, which assumes that the probability of purchase suddenly jumps as a particular number of friends acquire the product. The validation of these competing models is only made possible with sufficient data. We observed 16 million product recommendations between 4 million people on half a million products from a large online retailer. We found that probability of adoption follows a *diminishing returns* property, and that the probability of adoption saturates (and sometimes even starts to decrease) after around 20 network neighbors adopt [Leskovec et al., 2007a]. These findings are important for advertising and viral marketing.

Step 2 – Models: *How can we model information diffusion and cascades?* (Chapter 7)

We also study the information propagation and the cascades this process results in on the blogosphere. We analyzed one of the largest available collections of blog information, trying to find how blogs behave and how information propagates through the blogosphere. In contrast with viral marketing, stars and chains are basic components of blog cascades, with stars being more common.

Step 3 – Algorithms: *How can we effectively detect epidemics and disease outbreaks?* (Chapter 8)

The diminishing returns property has also led us to efficient and theoretically sound algorithms for network sensor placement [Leskovec et al., 2007c]. *Submodularity* is the diminishing returns property that we exploited to develop new tighter bounds for greedy optimization of submodular functions and to devise new efficient optimization algorithms. Our approach *provably* achieves *near optimal* placements, while being *700 times* faster on our dataset than a simple greedy algorithm. Our approach [Krause et al., 2008] ranked first in the “Battle of the Water Sensor Networks” competition where the task was to place sensors in a city water distribution network to effectively detect contaminants spreading over the network [Ostfeld et al., 2006]. Beyond the task at hand, we showed that the same sensor placement algorithm can be used to decide the best news sites on the internet to read to not miss important information, *i.e.*, to detect “information epidemics” effectively. We tracked the information propagation on the blogosphere for 1 year, and used our

algorithms to find the most informative blogs. Our project website <http://www.blogcascades.org> received more than 30,000 pageviews to date.

Contributions:

- Our work on the shape of the human adoption curve and cascades in viral marketing and blogosphere was the first to measure and analyze cascading behavior in a large real-world setting. We also found that the human adoption curve follows *diminishing returns*.
- We developed the CELF algorithm for sensor placement to detect disease outbreaks in networks. We proved that CELF placements are near optimal, and obtained data dependent bounds that show our solutions are at $\approx 90\%$ of NP-hard to compute optimal, while being *700 times faster* than a simple non-optimal greedy algorithm.

Impact:

- Our work on the CELF algorithm received the best student research paper award at ACM KDD 2007 conference [Leskovec et al., 2007c].
- Our approach for contamination detection in water distribution networks [Krause et al., 2008] ranked first in the “Battle of the Water Sensor Networks” competition where the task was to place sensors in a city water distribution network to effectively detect contaminants spreading over the network [Ostfeld et al., 2006].
- Follow-up works by Duncan Watts [Kossinets and Watts, 2006], Jon Kleinberg, Daniel Huttenlocher [Backstrom et al., 2006] and others later confirmed the diminishing returns behavior in a number of other domains, *e.g.*, the probability of joining a community, sending an email, or editing an article on Wikipedia.

1.2.3 Part 3 – Large data

The third part of the thesis presents our work on very large networks. We show how large amounts of data give us opportunities to observe phenomena that were previously practically invisible.

Step 1 – Observations: *What properties hold for a social network of the whole planet?* (Chapter 9)

We present the “planetary scale” Microsoft Instant Messenger network, the *largest social network* analyzed to date [Leskovec and Horvitz, 2008]. We collected and analyzed *4.5 terabytes* of network data. The MSN network contains *240 million* people, with more than *1 billion* conversations per day. We investigate on a planetary scale the oft-cited report that people are separated by “six degrees of separation” and find that the average path length among Messenger users is *6.6*. We also examine homophily and patterns of intra- and international conversation.

Step 2 – Models: *What is community or cluster structure of real-world networks?* (Chapter 10)

We present our work on community structure in networks. Researchers in the social sciences and physics have long been excited about the existence of “network communities”, where the intuition is that networks contain sets of nodes that interact more strongly with each other than with the remainder of the network. We found behaviors that are fundamentally different from intuitions based on small social networks, spatial graphs or hierarchical community structure that has typically been assumed for social and biological networks. Our observation is that, in large networks, tight communities exist only at smaller size scales.

The limit on the community size is ≈ 100 nodes which agrees well with Dunbar's observation that 150 is the maximum human community size [Dunbar, 1998]. As community exceeds this critical size it vanishes and blends with the rest of the network [Leskovec et al., 2008b]. Our observations were only possible since we examined large enough networks exceeding the size scale of communities. Formalization and models of such behavior would have a wide range of implications for researchers in the social sciences who want to discover communities from network data, and also for graph clustering and partitioning research [Leskovec et al., 2008b].

Step 3 – Algorithms: *How can we predict web search result quality without looking at the webpage content?* (Chapter 11)

Last, we present ways of how local web graph structure can be used for predicting the quality of web search results. We show how local structure of the web graph can be used to make globally accurate predictions about relevancy of web pages. We introduce *web projections*, where we extract context sensitive subgraphs of the web, and then use *machine learning* on contextual subgraphs of the web that can be used for *search result quality* prediction, *web spam* identification and predicting what search engine user will do next.

Contributions:

- We analyzed the properties of the planetary MSN Messenger social network, the *largest social network* examined up to date, and found the “*6.6 degrees of messaging*”, i.e., that people are on average separated by only 6.6 hops [Leskovec and Horvitz, 2008].
- Our analysis of community structure in large social and information networks showed that there is a maximum scale to a network community, which has many implications for clustering and community identification methods.

Impact:

- Our analysis on of MSN Messenger network, the largest network analyzed up to date, and the “*6.6 degrees of messaging*” appeared in popular press like Nature news, ZDNet, Cnet (all in March '08), Washington Post, MSNBC and BBC (all in August '08).
- Our work on the most influential bloggers generated lots of excitement as we experienced a “*Slash-dot effect*” with more than *30,000 visits* to our project website <http://www.blogcascades.org>. Moreover, the work also appeared in ACM TechNews (November '08) and on MSNBC (January '08).

Next, we present basic concepts and preliminaries, introduce the notation and briefly survey the related work. We then proceed with each of the three main parts of the thesis: Network evolution, Network cascades, and Large data.

Chapter 2

Overview and survey

In this chapter we review the basic concepts and terminology used in this thesis and introduce the notation. Next, we survey the works on properties of networks and models to explain their emergence, as well as network diffusion, cascading behavior and information propagation in networks.

2.1 Basic concepts and definitions

Next, we briefly define concepts and terminology that we will be using throughout the thesis. We introduce basic graph-theoretic concepts and review the power law distributions.

2.1.1 General graph-theoretic concepts

Network data is modeled or represented with a *graph*. A graph $G = (\mathcal{V}, \mathcal{E})$ is defined with a vertex set \mathcal{V} , where N denotes the number of nodes, $N = |\mathcal{V}|$, and an edge set \mathcal{E} , where E denotes the number of edges, $E = |\mathcal{E}|$. We interchangeably use terms vertex or node to refer to elements of the vertex set \mathcal{V} , and similarly edge, link or connection to refer to elements of the edge set \mathcal{E} .

A convenient way to represent a graph G is by using an *adjacency matrix*, which is an $N \times N$ matrix \mathbf{A} , where $\mathbf{A}_{i,j} = 1$ if $(i, j) \in \mathcal{E}$ and 0 otherwise.

Next we define the terminology and several basic graph-theoretic concepts:

Bipartite graph: graph G is bipartite if its vertex set can be partitioned into two disjoint sets $\mathcal{V}_1, \mathcal{V}_2$, so that there are only edges connecting nodes across the sets \mathcal{V}_1 and \mathcal{V}_2 . Or equivalently, there exist no edges between the nodes of the same partition.

Directed and undirected graph: A graph is *undirected* if $(i, j) \in \mathcal{E} \Leftrightarrow (j, i) \in \mathcal{E}$, *i.e.*, edges are unordered pairs of nodes. If pairs of nodes are ordered, *i.e.*, edges have direction, then the graph is *directed*.

Connectedness: We say that two nodes in a network are *connected* if there exists an undirected path between them.

Weakly and strongly connected graph: A graph is *connected* if there is a path between all pairs of nodes in a graph. If the graph is directed, then it is *weakly connected* if there exists an undirected path connecting any pair of nodes. Similarly graph is *strongly connected* if there exists a directed path connecting any pair of nodes in a graph.

Connected component: A *connected component* or just a component is a maximal set of nodes where for every pair of the nodes in the set there exist a path connecting them. Analogously, for directed graphs we have *weakly* and *strongly* connected components.

Biconnected graph: A graph is *biconnected* if the removal of any single edge does not disconnect the graph. This means that between any pair of nodes there exist at least 2 disjoint paths. Edges whose removal disconnects a connected graph are called *bridge edges*. Similarly, a node is an *articulation node* if its removal disconnects the graph.

Complete graph: A graph is complete if all pairs of nodes are connected.

Expander graph: An expander graph is a sparse graph which has high connectivity properties quantified using vertex (or edge) expansion: A graph G on N nodes is α -vertex expander if for any $\mathcal{S} \subset \mathcal{V}$ where $|\mathcal{S}| \leq N/2$ we have $|\delta(\mathcal{S})| \geq \alpha|\mathcal{S}|$. Here $\delta(\mathcal{S})$ denotes a set of all edges with one end in \mathcal{S} and the other end in $\mathcal{V} \setminus \mathcal{S}$.

Loosely speaking, G is an expander if α is “large”. Intuitively, an expander is a graph for which any “small” subset of vertices has a relatively “large” neighborhood, or similarly, removing random edges does not reduce the property of an expander by much.

Subgraph: A *subgraph* $G_s = (\mathcal{V}_s, \mathcal{E}_s)$ of a graph $G = (\mathcal{V}, \mathcal{E})$ is a subset of edges and all their endpoints: $\mathcal{E}_s \subseteq \mathcal{E}$ and $\mathcal{V}_s = \{i, j : (i, j) \in \mathcal{E}_s\}$.

Induced subgraph: An *induced subgraph* $G_s = (\mathcal{V}_s, \mathcal{E}_s)$ of a graph $G = (\mathcal{V}, \mathcal{E})$ is a subset of nodes and all their edges: $\mathcal{V}_s \subseteq \mathcal{V}$ and $\mathcal{E}_s = \{(i, j) : (i, j) \in \mathcal{E} \wedge i, j \in \mathcal{V}_s\}$.

Node degree: We say that a node has degree d if it has d incident nodes. For directed graphs we talk about out-degree d_{out} , which is the number of edges pointing from the node. Similarly, in-degree d_{in} denotes the number of edges pointing towards the node. For undirected graphs for every node u $d_{in}(u) = d_{out}(u) = d(u)$. We also define the graph average degree $\bar{d} = 1/N \sum_{u \in \mathcal{V}} d(u) = 2E/N$.

Triad: A triad (or a triangle) is a triple of connected nodes (u, v, w) , *i.e.*, $(u, v), (v, w), (w, u) \in \mathcal{E}$.

2.1.2 Diameter and effective diameter

For each natural number h , let $g(h)$ denote the fraction of connected node pairs whose shortest connecting path has length at most h , *i.e.*, at most h hops away. The *hop-plot* for the network is the set of pairs $(h, g(h))$; it thus gives the cumulative distribution of distances between connected node pairs. We extend the hop-plot to a function defined over all positive real numbers by linearly interpolating between the points $(h, g(h))$ and $(h + 1, g(h + 1))$ for each h , and we define the *effective diameter* of the network to be the value of h at which the function $g(h)$ achieves the value 0.9.

Definition 2.1.1. *Graph G has the diameter D if the maximum length of undirected shortest path over all connected pairs of nodes is D . The length of the path is the number of segments (edges, links, hops) it contains.*

We also use *full diameter* to refer to this quantity. Notice the difference between the usual and our definition of the diameter. For a disconnected graph the diameter as usually defined to be infinite, here we avoid this problem by considering only pairs of nodes that are connected. Also note we ignore the directionality of an edge if the graph is directed.

Definition 2.1.2. *For each natural number h , let $g(h)$ denote the fraction of connected node pairs whose undirected shortest connecting path in a graph G has length at most h . And let D' be an integer for which $g(D' - 1) < 0.9$ and $g(D') \geq 0.9$. Then the graph G has the integer effective diameter D' [Tauro et al., 2001].*

In other words, the *integer effective diameter* is the smallest number of hops D' at which at least 90% of all connected pairs of nodes can be reached.

Last we give the definition of the *effective diameter* as considered in this thesis. Originally we defined $g(h)$, a fraction of connected pairs of nodes at distance at most h , only for natural numbers h . Now we extend the definition of g to all positive reals x by linearly interpolating the function value between $g(h)$ and $g(h + 1)$ ($h \leq x < h + 1$): $g(x) = g(h) + (g(h + 1) - g(h))(x - h)$.

Definition 2.1.3. *Let D^* be a value where $g(D^*) = 0.9$, then graph G has the effective diameter D^* .*

This definition varies slightly from an alternate definition of the effective diameter used in earlier work: the minimum integer value h such that at least 90% of the connected node pairs are at distance at most h . Our variation smoothes this definition by allowing it to take *non-integer* values.

The effective diameter is a more robust quantity than the diameter (defined as the maximum distance over all connected node pairs), since the diameter is prone to the effects of degenerate structures in the graph (e.g., very long chains). However, our experiments show that the effective diameter and diameter tend to exhibit qualitatively similar behavior. Note that under these definitions the effective diameter and the diameter are well defined even if the graph is disconnected.

Calculating the exact diameter or effective diameter is infeasible for large networks at it takes $O(N^3)$ time. One way to overcome this would be to resort to sampling, *i.e.*, sample a large number of node pairs and calculate the length of the shortest paths between pairs. We chose a different approach, and rather used an approximation algorithm ANF [Palmer et al., 2002] that is based on fast approximate counting and hashing.

2.1.3 Power law distributions and heavy tails

Here we describe the power law and heavy-tailed distributions and then make connections to several network properties that usually follow power law distributions. Further details on mathematics of power laws can be found in [Mitzenmacher, 2004, Newman, 2005, Clauset et al., 2007].

A distribution is a Power law if it has a PDF (probability density function) of the form

$$p(x) \propto x^{-\gamma}$$

where $p(x)$ is the probability to encounter value x and γ is the exponent of the power law.

If x is a continuous random variable then $p(x)dx = Pr(x \leq X < x + dx) = \frac{1}{Z}x^{-\gamma}dx$, where Z is a normalizing constant. The density diverges as $x \rightarrow 0$ so the equation cannot hold for all x ; so there must

be some lower bound x_{min} to power law behavior. Provided that $\gamma > 1$ then calculating the normalizing constant we find that:

$$p(x) = \frac{\gamma - 1}{x_{min}} \left(\frac{x}{x_{min}} \right)^{-\gamma}$$

For the case when x is discrete and takes integer values we obtain $p(x) = Pr(X = x) = \frac{1}{Z} x^{-\gamma}$. Again the distribution diverges at zero, so there must be a lower bound x_{min} on the power law behavior. Calculating the normalizing constant we find that

$$p(x) = \frac{x^{-\gamma}}{\zeta(\gamma, x_{min})}$$

where $\zeta(\gamma, x_{min}) = \sum_{i=0}^{\infty} (i + x_{min})^{-\gamma}$ is the generalized zeta function.

In many cases it is useful to consider the Complementary Cumulative Distribution Function (CCDF) of a power law distributed random variable. In both discrete and continuous case it is defined as $Pr(X \geq x)$. For continuous case $Pr(X \geq x) = (\frac{x}{x_{min}})^{-\gamma+1}$ and for discrete case $Pr(X \geq x) = \frac{\zeta(\gamma, x)}{\zeta(\gamma, x_{min})}$.

Perhaps surprisingly, power law distributions can have infinite variances and even the mean can be infinite. Basically, one can show that for a power law distribution with power law exponent γ moments $m < \gamma - 1$ will exist and all higher moments will diverge. For example, for $\gamma \leq 2$ mean, variance and all other moments are infinite; similarly, for $2 < \gamma \leq 3$ mean exists (*i.e.*, is finite), while variance will be infinite, and for $\gamma > 3$ mean and variance will be finite, while third and all higher moments will diverge.

Heavy-tailed and scale-free distributions

A power law distribution is sometimes called a *scale-free* distribution, which intuitively means that it looks the same regardless of what scale we look at it on. More precisely,

Definition 2.1.4. *Distribution $p(x)$ for a quantity x is scale-free if there exists a function $g(b)$ such that $p(bx) = g(b)p(x)$ for all b and x .*

The scale-free property means that when we increase the scale or units by which we measure x by factor b the shape of the distribution $p(x)$ is unchanged except for the multiplicative constant. This means that no matter what range of x one looks at, the proportion of small to large events is the same, *i.e.*, the slope of the curve on any section of the log-log plot is the same.

Exponential family distributions (like Gaussian distribution) are not scale-free. Actually, power law is the only scale-free distribution [Newman, 2005].

Similarly, power law is also a heavy-tailed distribution. This means that its tails are not exponentially bounded; that is, they have heavier tails than the exponential distribution. More precisely, we define heavy-tails in the following way [Asmussen, 2003]:

Definition 2.1.5. *The distribution of a random variable X is heavy-tailed if*

$$\lim_{x \rightarrow \infty} \frac{Pr(X > x)}{e^{-\epsilon x}} = \infty$$

for all $\epsilon > 0$.

In contrast, we say a distribution is light-tailed if the limit < infinity for some ϵ .

Examples of heavy-tailed distributions include power law distributions, Pareto and others which we examine next.

Relation to Zipf and Pareto distributions

Zipf's law [Zipf, 1949] usually refers to the rank-frequency plots, *i.e.*, “size” or magnitude y of an occurrence of an event relative to its rank r . Zipf's law is named after George Kingsley Zipf, a Harvard linguistics professor, who tried to determine the “size” of the r^{th} most common English word. Size here denotes the frequency of use of the word in English text. Zipf's law states that the size of the r^{th} largest occurrence of the event is inversely proportional to its rank: $y \propto r^{-b}$ with $b \approx 1$.

Pareto distribution is named after economist Vilfredo Pareto, who was interested in the distribution of income [Lorenz, 1905]. Instead of asking what the r^{th} largest income is, Pareto asked how many people have an income greater than x . Pareto's law is given in terms of the complementary cumulative distribution function (CCDF), *i.e.*, the number of events larger than x is an inverse power of x : $\Pr(X \geq x) \propto x^{-k}$. Basically, it states that there are a few multi-billionaires, but most people make only a modest income. When this distribution is used to model the distribution of wealth, then the parameter k is called the Pareto index. In 1906 Pareto also made the observation that twenty percent of the population owned eighty percent of the property in Italy, *i.e.*, the 80 – 20 rule (that occurs for power law exponent $\gamma = 2$).

Interestingly, power law, Pareto distributions and the Zipf's law are all intimately related. Relation between the power law scaling exponent γ and the Zipf's law exponent b is $\gamma = 1 + (1/b)$. Similarly for the relation of power law exponent γ and Pareto index k we obtain $\gamma = k + 1$ [Adamic, 2000].

Estimating power law parameters from empirical data

Most commonly the parameters of the power law distribution are estimated from a simple histogram. Taking logs on both sides of the power law equation gives $\ln p(x) = \gamma \ln x + \text{const}$, which implies that a histogram follows a straight line when plotted on log–log scales. We calculate the empirical probability density function of x (histogram of its frequency distribution) and plot the histogram on log-log axis. If a distribution approximately follows a straight line, then one could assert that distribution follows a power law with exponent γ given by the slope of the line. Unfortunately this method shows some bias as the independence and Gaussian noise assumption of least squares linear regression are violated [Newman, 2005].

A better but still not entirely correct way of parameter estimation is by fitting a straight line on a log-log plot of the CCDF. This gives less biased results as the visual form of the CCDF is more robust against fluctuations due to finite sample sizes [Clauset et al., 2007]. To improve the accuracy one also bins the data using the exponentially increasing bin widths, *i.e.*, logarithmic binning.

The Maximum Likelihood Estimates (MLE) are unbiased. For the continuous case of power law distribution for the power law exponent γ the MLE is:

$$\hat{\gamma} = 1 + n \left[\sum_{i=1}^n \ln \frac{x_i}{x_{\min}} \right]^{-1}$$

where x_i , $i = 1, \dots, n$, are the observed values x such that $x \geq x_{\min}$.

For the discrete case there is no closed form solution for the MLE estimate of the power law exponent. The most convenient way to estimate γ is to directly optimize the log-likelihood function:

$$l(\gamma) = -n \ln \zeta(\gamma, x_{\min}) - \gamma \sum_{i=1}^n n \ln x_i$$

Empirically MLE estimates work best and give unbiased results. However many times they give visually unsatisfying estimates, especially as one has to estimate also the start of the power law tail x_{min} which in practice is hard to estimate [Clauset et al., 2007].

2.2 Statistical properties of networks

Networks are composed of nodes and edges connecting them. Depending on the domain network data comes from they can be represented by directed or undirected networks. Examples of networks include the Internet, World Wide Web, social networks of acquaintance, collaboration or other connections between individuals, organizational networks, metabolic networks, language networks, food webs, distribution networks such as water distribution networks, blood vessels or postal delivery routes, networks of citations between papers, software networks where edges represent dependencies or function calls.

Research over the past few years has identified classes of properties that can be found in many real-world networks from various domains. While many patterns have been discovered, two of the principal ones are heavy-tailed degree distributions and small diameters.

Degree distributions: The degree-distribution of a graph is a power law if the number of nodes N_d of degree d is given by $N_d \propto d^{-\gamma}$ ($\gamma > 1$) where γ is called the *power law degree exponent*.

Such degree distributions have been identified in phone call graphs [Abello et al., 1998], the Internet [Faloutsos et al., 1999], the Web [Kleinberg et al., 1999, Broder et al., 2000, Barabási and Albert, 1999, Huberman and Adamic, 1999, Kumar et al., 1999b], citation graphs [Redner, 1998], online social networks [Chakrabarti et al., 2004], click-stream data [Bi et al., 2001], and many others.

Typically for most datasets the degree exponent γ takes values $2 < \gamma < 3$. For example, in-degree distribution of web graph has $\gamma_{in} = 2.1$ and out-degree $\gamma_{out} \approx 2.4$ [Albert and Barabási, 2002], while Autonomous systems have $\gamma \approx 2.4$ [Faloutsos et al., 1999]. However, deviations from the power law pattern have been noticed [Pennock et al., 2002], which can be explained by the “DGX” distribution [Bi et al., 2001].

Most of large real-world networks have heavy-tailed or power law degree distributions, and are thus often called scale-free networks. This discovery [Faloutsos et al., 1999] is important as it shows that real networks are not “random” (as we will more precisely define below). Moreover, in scale-free networks there are many vertices with a degree that greatly exceeds the average (a direct result of power law degree distributions). These highest-degree nodes are often called “hubs”, and are thought to serve specific purposes in their networks, although this depends greatly on the domain.

The notion of self-similarity is implied in the name “scale-free”. Intuitively, a self-similar object consists of miniature replicas of itself [Schroeder, 1991]. Several researchers have argued that especially web graphs [Dill et al., 2002, Dorogovtsev et al., 2002, Crovella and Bestavros, 1997] and biological networks [Ravasz and Barabási, 2003] tend to be self-similar and “fractal”.

Small diameter: Most real-world graphs exhibit relatively small diameter, which is also known as the “small-world” phenomenon: A graph has diameter d if every pair of nodes can be connected by a path of length at most d . The diameter d is susceptible to outliers. Thus, a more robust measure of the pairwise distances between nodes of a graph is the *effective diameter* [Tauro et al., 2001] as we defined it in definition 2.1.3. The effective diameter has been found to be small for large real-world graphs,

like Internet, Web, and social networks [Albert and Barabási, 2002, Milgram, 1967, Albert et al., 1999, Bollobas and Riordan, 2004, Broder et al., 2000, Chung and Lu, 2002a, Watts and Strogatz, 1998]).

Scree plot: This is a plot of the eigenvalues (or singular values) of the graph adjacency matrix of the graph, versus their rank, using a log-log scale. The scree plot for real networks is often found to approximately obey a power law [Dorogovtsev et al., 2002, Faloutsos et al., 1999]. The distribution of components of the elements of the first eigenvector (indicators of “network value”) has also been found to be skewed following a power law distribution [Chakrabarti et al., 2004].

Triads and clustering coefficient: Clustering coefficient is a measure of transitivity in networks and especially in social networks [Watts and Strogatz, 1998], *i.e.*, friend of a friend is more likely to be also my friend. In many networks it is found that if node u is connected to v and v is further connected to w then there is a higher probability that node u is connected to w . In terms of network topology, transitivity means the presence of a heightened number of triangles in the network, *i.e.*, sets of fully connected triples of nodes.

Clustering coefficient C_d of a vertex of degree d is defined as follows. Let node v have d neighbors; then at most $d(d - 1)/2$ edges can exist between them. Let C_v denote the fraction of these allowable edges that actually exist. This basically means that clustering coefficient C_v of a vertex v is the proportion of links between the vertices within its neighborhood divided by the number of links that could possibly exist between them. Or equivalently, C_v is the fraction of triangles (triads) centered at node v among the $d(d - 1)/2$ triangles that could possibly exist. Then C_d is defined as the average C_v over all nodes v of degree d , and the global clustering coefficient C is the average C_v over all nodes v .

It has been found that clustering coefficient in real networks is significantly higher than for random networks (conditioned on same degree distribution). Moreover, it has been also observed [Dorogovtsev et al., 2002, Ravasz and Barabási, 2003] that in real networks clustering coefficient C_d decreases as the node degree d increases. Moreover, C_d scales as a power law, $C_d \propto d^{-1}$. This observation has been somewhat quickly used as an indication of hierarchical network organization [Ravasz et al., 2002, Ravasz and Barabási, 2003].

The idea is that the low-degree nodes belong to very dense sub-graphs and those sub-graphs are connected to each other through hubs. Consider a social network in which nodes are people and links are acquaintance relationships between people. People tend to form communities, *i.e.*, small groups in which everyone knows almost everyone else; and such groups can then be hierarchically nested or organized. In addition, the members of a community also have a few acquaintance relationships to people outside that community.

A variant of clustering coefficient for directed graphs has also been defined and examined by Ahnert and Fink [Ahnert and Fink, 2008]. Authors found that different types of networks have various kinds triangles more expressed. For example, feed forward loops are very common in transcription networks, while cycles are most common in language networks.

Community structure: A large body of work has been devoted to defining and identifying communities in social and information networks. Communities, modules or clusters are most often thought as sets of nodes that has more and/or better-connected edges between its members than between members of that set and the remainder of the network [Radicchi et al., 2004, Girvan and Newman, 2002]. Many times it is also naturally assumed that the communities observe a recursive structure, where bigger communities can further be split into smaller and smaller communities [Clauset et al., 2006, Sales-Pardo et al., 2007].

The problem of community identification is often formulated as unsupervised learning, some form of clustering or graph partitioning where the idea is to partition the network into disjoint but sometime also overlapping sets of nodes, where there few edges need to be cut to separate internally densely linked set of nodes, *i.e.*, a community. For example, see the reviews on community identification [Newman, 2004, Danon et al., 2005, Palla et al., 2005, Clauset et al., 2008], data clustering [Jain et al., 1999], and graph and spectral clustering [Gaertler, 2005, Schaeffer, 2007, von Luxburg, 2006].

It has been observed that community-like sets of nodes tend to correspond to organizational units in social networks [Newman, 2006b], functional modules in biological networks [Ravasz et al., 2002], and scientific disciplines in collaboration networks between scientists [Girvan and Newman, 2002].

A somewhat contrary concept to hierarchical community structure is the “core-periphery” structure of the network [Borgatti and Everett, 2000, Holme, 2005], that in computer science also goes by the name of the *jellyfish* [Siganos et al., 2006], or the *octopus* [Chung and Lu, 2006a] structure of the network. All of the above basically say that the network is composed of a large and densely interlinked network core that basically has no community structure. The remainder of the nodes is a part of the periphery, where the periphery nodes have links towards the core, but are not connected among themselves.

Core-periphery structure suggests the opposite of the community structure or the hierarchical network structure. In core-periphery there is a densely linked and intermingled network core, and a number of nodes on the periphery with their links pointing towards the core.

For example, Internet Autonomous Systems [Siganos et al., 2006] have been found to have this structure. And as we will later see in Chapter 10 this network structure is present in almost all large networks with more than several thousand nodes.

Network motifs: Network motifs [Milo et al., 2002, Alon, 2007] are basic building blocks of complex networks. They are of interest in gene regulatory and other biological networks, like protein-protein interaction networks, signal transduction networks and metabolic networks [Shai et al., 2002].

The idea is to enumerate and count occurrences of all possible induced subgraphs of a given graph G up to a small number of nodes. Usually, subgraphs of size up to $N \leq 5$ nodes, as the combinatorial explosion of the number of possible graphs and the graph isomorphism test that is needed when counting make the computation unfeasible for larger N .

The frequencies of motifs are then compared to those of a random graph conditioning on the same degree distribution. (See [Milo et al., 2004] for how to generate such graphs.) This way a random graph with same degree distribution is taken as a null-model and motifs that occur significantly more frequently in real graph than in the null-model are then extracted. Different studies have argued that certain motifs are found frequently in biological networks, and then tried to assign them a biological function.

For example, a node with a self-loop is the simplest possible motif in a regulatory network. It is called the autoregulation motif, and it has been argued that it controls for up-regulation or down-regulation of its own expression/activity. It has been shown that this motif appears at least 40 times in the *E. coli* regulatory network [Shai et al., 2002], which is much greater than what is expected by chance. Moreover, other motifs, like feed forward or feed backward loop, and have also been assigned biological functions.

Network motifs are interesting as they are exploring the basic building blocks from which networks are composed. In chapters 6 and 7 we will observe the cascading behavior in viral marketing and blogosphere, and we will present similar analysis of cascade motifs, *i.e.*, what do network cascades look like and what are their building blocks.

Additional network properties: Apart from these, several other patterns have been found in networks. For example, the “resilience” [Albert and Barabási, 2002, Palmer et al., 2002] shows that real-networks are resilient to random node attacks, *i.e.*, one can remove many randomly selected nodes from the network and the connectivity is not impacted by much. However, if one performs a targeted attack by removing just a few high degree hub nodes, the network connectivity gets severely disrupted. Other properties are also “stress” [Chakrabarti et al., 2004], network navigation [Kleinberg, 1999b, Watts et al., 2002], and many more.

We point the reader to [Albert and Barabási, 2002, Newman, 2003, Li et al., 2005, Boccaletti et al., 2006, Chakrabarti and Faloutsos, 2006] for overviews of the structural properties of networks. The book on social network analysis [Wasserman and Faust, 1994] is also useful reading.

2.3 Models of network structure and evolution

In parallel with empirical studies of large networks, there has been considerable work on models for graph generation. Both deterministic and stochastic models have been explored. Most often the models do not “force” the network to have a certain property but rather give general principles or mechanisms of edge creation that consequently lead to the global statistical property or distribution to arise in the network.

Erdős–Rényi random graph model

The earliest probabilistic generative model for graphs was a random graph model introduced by Erdős and Rényi [Erdős and Rényi, 1960]. The model states that given a number of nodes each pair of nodes has an identical, independent probability of being joined by an edge. There are two variants of the model: $G_{n,p}$ is defined to have n nodes, and each edge appears independently with probability p . Similarly, the $G_{n,m}$ is defined to have n nodes and m uniformly at random placed edges. There exists a close correspondence between the models, as in practice most theorems hold for both variants.

The study of Erdős–Rényi random graph model has led to a rich mathematical theory. For example, one can study the evolution of $G_{n,m}$, where one starts with the empty graph on n nodes and then keeps adding random edges one at a time. The graph will then be a $G_{n,m}$ where m is the number of edges added so far, *i.e.*, if one draws $G_{n,m}$ at random and adds a random edge the new graph will be $G_{n,m+1}$.

In evolution of $G_{n,m}$ there exists sharp thresholds or *phase transitions* in emergence of certain network properties. For example, there is a sharp threshold for the size of the largest connected component. Let $\bar{d} = 2m/n$ denote the average degree, then if $\bar{d} = 1 - \epsilon$ then graph is disconnected and all components are of size $O(\log n)$. When $\bar{d} = 1 + \epsilon$ there is exactly one component of size $\Omega(n)$, *i.e.* the giant component, and all other components are of size $O(\log n)$. This is exactly the point, the threshold, where the giant connected component emerges. Moreover, one can also prove that all other components are just trees plus one edge so they have at most one cycle [Bollobas and Riordan, 2003].

Similarly, one can show that degree distribution of Erdős–Rényi random graph follows a binomial distribution with mean \bar{d} [Albert and Barabási, 2002]. Moreover, the diameter (longest shortest path) of a random graph increases with the number of nodes n as $O(\log n)$, and the average shortest path length grows as $O(\log \log n)$ [Chung and Lu, 2001].

There is a rich mathematical theory about this model; however, the model is not realistic as it produces graphs that fail to match real-world networks in a number of respects (*e.g.*, it does not produce power law degree distributions).

Preferential attachment

The discovery of degree power laws led to the development of random graph models that exhibited such degree distributions, including the family of models based on *preferential attachment* [Barabási and Albert, 1999, Cooper and Frieze, 2003]. The model operates in the following way. Nodes are arriving one at a time. And when a new node u arrives to the network it creates m edges (m is a parameter and is constant for all nodes). The edges are not placed uniformly at random but preferentially, *i.e.*, probability that a new edge of u is placed to a node v of degree $d(v)$ is proportional to its degree, $p_u(v) \propto d(v)$.

This model was first described by Herb Simon [Simon, 1955] and he uses the term Yule distribution to refer to the power law distribution. Empirically power law degree distributions were first discovered in citation networks by D.J. de Solla Price [de Solla Price, 1965], where Price noticed that the number of new citations a paper obtains is proportional to the current number of citations. He calls this the “cumulative advantage” or the “rich get richer” phenomenon.

This simple behavior leads to power law degree tails with exponent $\gamma = 3$. Moreover it also leads to low diameters. The diameter in preferential attachment model grows slowly, *i.e.*, logarithmically with the number of nodes [Lu, 2001]. More precisely, diameter grows as $\log(N)$ when a new node adds a single edge ($m = 1$), and as $\log(N)/\log\log(N)$ for $m \geq 2$.

There are also many extensions to the Preferential attachment model. We mention three of them: the fitness model, Winners don’t take all, and the geometric preferential attachment.

In Preferential attachment model nodes that arrive early will end up having highest degrees. However, one could envision that each node has an inherent competitive factor that nodes may have, capable of affecting the network’s evolution. This is called node *fitness* [Bianconi and Barabási, 2001, Dorogovtsev et al., 2000, Ergün and Rodgers, 2002]. The idea is that intrinsic ability of a node to attract links in the network varies from node to node. The most efficient (or “fit”) nodes are able to gather more edges at the expense of others. In that sense, not all nodes are identical, and they claim their degree increase in the number of edges accordingly to the fitness they possess every time. Fitness parameter is usually considered as not varying over time and is multiplicative to the edge probability.

In spirit similar is the *Winners don’t take all* [Pennock et al., 2002] model where the intuition is taken from the web. It has been observed that for web communities of interest the distribution of links no longer follows a power law but rather resembles a normal distribution [Pennock et al., 2002]. Based on this observation, the authors then propose a generative model that mixes preferential attachment with a baseline probability of gaining a link.

A last variant of Preferential Attachment that we also describe is the *Geometric Preferential Attachment* [Flaxman et al., 2004, 2007], where the idea is to incorporate geography into the Preferential Attachment model. Intuition is that probability of linking to a node of degree d should be higher if the node is closer rather than farther. In this model nodes belong to some underlying geometry and then each node connects preferentially to other nodes inside some local ball of radius r . For example, one can scatter nodes uniformly on a sphere, and each node uses Preferential Attachment mechanism to attach to other nodes in some local neighborhood as defined by the sphere.

Copying model

Similar in spirit to the above models is the *copying model* [Kleinberg et al., 1999, Kumar et al., 2000], where a new node joins the network by uniformly at random choosing node u and then either linking to u 's neighbors or creating a random edges. More precisely, nodes are arriving one at a time. A new node v chooses k , the number of edges to add, and then with probability β it selects k vertices uniformly at random and links to them; and with probability $1 - \beta$ node v links to k random neighbors of a uniformly at random chosen node u , i.e., v copies u 's links. Copying model generates power law degree distributions with exponent $\gamma = 1/(1 - \beta)$.

There are also many related models where a new node selects an existing node u and then starts a random walk or breath first search type of procedure to create links to nodes in u 's vicinity. Such models include the *growing network with copying* model [Krapivsky and Redner, 2005], *Recursive search model* [Vazquez, 2001], and the *Random Surfer Model* [Blum et al., 2006], that is based on starting a random walk from node u and after each step restarting or with some probability creating a link.

Other models of scale-free networks

There are many other ways to explain the emergence of scale-free networks. For example, *Heuristically optimized tradeoffs* [Fabrikant et al., 2002] and *Highly optimized tolerance* [Carlson and Doyle, 1999, Doyle and Carlson, 2000, 2002] are two models where power law degree distributions emerge as a result of optimization. For example, on the internet one wishes to maximize the connectivity (ping time), while minimizing the cost of the physical connection. Power laws naturally emerge in such case [Fabrikant et al., 2002].

Alternative models for generating scale-free networks with power law degree distributions include *configuration model* [Bollobas, 1980, Aiello et al., 2000, Bollobas and Riordan, 2003], where nodes have a number of outward pointing spokes and then these spokes are connected uniformly at random. This closely resembles the Erdős–Rényi random graph model so many tools developed for analysis of random graphs apply. The distribution of a number of spokes of a node defines degree distribution of a graph. Chung and Lu [Lu, 2001] proposed a different model where node degree sequence w_i is generated (e.g., sampled from power law distribution) and the edge (u, v) appears with probability $w_u \cdot w_v / \sum_i w_i$. In this model the expected degree sequence will follow the sequence w_i .

Small-world model

Last family of network models we describe here strives for small diameters and local structures, like triangles, in networks that arise from geographical proximity or homophily. Such models include the *small-world* model [Watts and Strogatz, 1998] and the Waxman generator [Waxman, 1988]. In a small-world model one starts with a regular lattice (e.g., a grid). The lattice models local short-range links. Then for each edge with probability p we move its endpoint to a uniformly at random chosen node. The model offers a nice way of interpolating between regular ($p = 0$) and random graphs ($p = 1$). For low p graphs will have lots of local structure with many short range links, clustering will be high but the diameter will be also large. As one increases p long range edges will start to appear which will have the effect to destroy the local structure (clustering will decrease) but at the same time the diameter of the network will also decrease.

Related to the small-world is the concept of “navigability” or “searchability” in networks [Kleinberg, 1999b] where the question is how to locally route a message to a target node so that it reaches the target as quickly as possible. In fact, it has been shown that the structure of real networks allows local routing and navigation [Liben-Nowell et al., 2005].

For a more extensive review of the topic of network models and generators we point the reader to recent works [Albert and Barabási, 2002, Chakrabarti and Faloutsos, 2006, Bollobas and Riordan, 2003, Newman, 2003] that give a survey of the structural properties and statistics of real world graphs and the underlying generative models for graphs.

2.4 Diffusion and cascading behavior in networks

Information cascades are phenomena in which an action or idea becomes widely adopted due to influence by others [Bikhchandani et al., 1992]. Cascades are also known as “fads” or “resonance.” Cascades have been studied for many years by sociologists concerned with the *diffusion of innovation* [Rogers, 1995]; more recently, researchers in several fields have investigated cascades for the purpose of selecting trendsetters for viral marketing [Domingos and Richardson, 2001], finding inoculation targets in epidemiology [Newman et al., 2002], and explaining trends in blogosphere [Kumar et al., 2003]. Despite much empirical work in the social sciences on datasets of moderate size, the difficulty in obtaining data has limited the extent of analysis on very large-scale, complete datasets representing cascades. We look at the patterns of influence in a large-scale, real recommendation network and examine the topological structure of cascades.

Most of the previous research on the flow of information or influence through the networks has been done in the context of epidemiology and the spread of diseases or viruses over the network [Bailey, 1975, Anderson and May, 2002]. Classical disease propagation models are based on the stages of a disease in a host: a person is first *susceptible* to a disease, then if she is exposed to an infectious contact she can become *infected* and thus *infectious*. After the disease ceases the person is then either *recovered* or *removed*. After that a person becomes *immune* for some period. The immunity can also wear off and the person becomes again susceptible. Thus SIR (susceptible – infected – recovered) models diseases where a recovered person never again becomes susceptible, while SIRS (SIS, susceptible – infected – (recovered) – susceptible) models population in which recovered host can become susceptible again. Given a network and a set of infected nodes the *epidemic threshold* is studied, *i.e.*, conditions under which the disease will either dominate or die out.

Diffusion models that try to model the process of adoption of an idea or a product can generally be divided into two groups:

- *Threshold model* [Granovetter, 1978] where each node in the network has a threshold $t \in [0, 1]$, typically drawn from some probability distribution. We also assign *connection weights* $w_{u,v}$ on the edges of the network. A node adopts the behavior if a sum of the connection weights of its neighbors that already adopted the behavior (purchased a product in our case) is greater than the threshold: $t \leq \sum_{\text{adopters}(u)} w_{u,v}$.
- *Independent cascade model* [Goldenberg et al., 2001] where whenever a neighbor v of node u adopts, then node u also adopts with probability $p_{u,v}$. In other words, every time a neighbor of u purchases a product, there is a chance that u will decide to purchase as well.

While these models address the question of how influence spreads in a network, they are based on *assumed* rather than *measured* influence effects. In contrast, our study tracks the actual diffusion of recommendations through email, allowing us to quantify the importance of factors such as the presence of highly connected individuals, or the effect of receiving recommendations from multiple contacts. Compared to previous empirical studies which tracked the adoption of a single innovation or product, our data encompasses over half a million different products, allowing us to model a product's suitability for viral marketing in terms of both the properties of the network and the product itself.

2.4.1 Information cascades in blogosphere

Most work on extracting information cascades has been done in the blog domain [Adamic and Glance, 2005, Adar and Adamic, 2005, Gruhl et al., 2004]. The authors in this domain noted that, while information propagates between blogs, examples of genuine cascading behavior appeared relatively rarely. This is possibly due to bias in the web-crawling and text analysis techniques used to collect pages and infer relationships. In our dataset, all the recommendations are stored as database transactions, and we know that no records are missing. Associated with each recommendation is the product involved, and the time the recommendation was made. Studies of blogosphere either spend a lot of effort mining topics from posts [Adar and Adamic, 2005, Gruhl et al., 2004] or consider only the properties of blogosphere as a graph of unlabeled post or blog URLs [Adamic and Glance, 2005].

There are several potential models to capture the structure of the blogosphere. Work on information diffusion based on topics [Gruhl et al., 2004] showed that for some topics, their popularity remains constant in time (“chatter”) while for other topics the popularity is more volatile (“spikes”). [Kumar et al., 2003] analyze community-level behavior as inferred from blog-rolls — permanent links between “friend” blogs. In their extension [Kumar et al., 2006] performed analysis of several topological properties of link graphs in communities, finding that much behavior was characterized by star like graph structure, *i.e.*, a single charismatic individual linked to many users each with very few other connections.

2.4.2 Cascades in viral marketing

Viral marketing can be thought of as a diffusion of information about the product and its adoption over the network. Primarily in social sciences there is a long history of research on the influence of social networks on innovation and product diffusion. However, such studies have been typically limited to small networks and typically a single product or service. For example, [Brown and Reingen, 1987] interviewed the families of students being instructed by three piano teachers, in order to find out the network of referrals. They found that strong ties, those between family or friends, were more likely to be activated for information flow and were also more influential than weak ties [Granovetter, 1973] between acquaintances.

In the context of the internet, word-of-mouth advertising is not restricted to pairwise or small-group interactions between individuals. Rather, customers can share their experiences and opinions regarding a product with everyone. Quantitative marketing techniques have been proposed [Montgomery, 2001] to describe product information flow online, and the rating of products and merchants has been shown to effect the likelihood of an item being bought [Resnick and Zeckhauser, 2002, Chevalier and Mayzlin, 2006]. More sophisticated online recommendation systems allow users to rate others’ reviews, or directly rate other reviewers to implicitly form a trusted reviewer network that may have very little overlap with a person’s actual social circle. [Richardson and Domingos, 2002b] used Epinions’ trusted reviewer network

SYMBOL	DESCRIPTION
G	Graph or graph adjacency matrix
G_t	Graph composed of nodes and edges that arrived before time t
N	Number of nodes in a graph
E	Number of edges in a graph
$N(t)$	Number of nodes in a graph at time t
$N(e)$	Number of nodes in a graph at time t
u, v, w	Nodes in a graph
$e = (u, v)$	Edge in a graph
$d(u)$	Degree of node u (number of edges incident to node u)
d	Degree
\bar{d}	Average node degree in a graph
d_{max}	Maximum node degree in a graph
γ	Power law degree exponent, $p(d) \propto d^{-\gamma}$
a	Densification power law exponent, $E(t) \propto N(t)^a$
$h(u, v)$	Length of the shortest path between nodes u and v
h	Number of hops, path length, distance
D	Diameter of a graph as defined in 2.1.1
D^*	Effective diameter of the graph as defined in 2.1.3
\mathcal{A}	Set of elements, $\mathcal{A} = \{a_1, \dots, a_n\}$

Table 2.1: Table of common symbols.

to construct an algorithm to maximize viral marketing efficiency assuming that individuals' probability of purchasing a product depends on the opinions on the trusted peers in their network. [Kempe et al., 2003] have followed up on the challenge of maximizing viral information spread by evaluating several algorithms given various models of adoption we discuss next.

2.5 Table of symbols

We list common symbols used in the thesis. Each chapter then also defines chapter-specific concepts and symbols. For the comprehensive list of symbols refer to the appendix table A.1.

2.6 Table of datasets

In this thesis we use more than 100 different network datasets. Tables A.2, A.3, and A.4 give brief descriptions and some of the basic statistics, like number of nodes and edges, diameter, clustering coefficient and so on.

Part I

Network evolution

**How do large networks evolve and
how to model this?**

Part 1 – Network evolution: Overview

Networks, especially social networks and the web, are not static but evolve over time by additions and deletions of nodes and edges. Here we examine such evolutionary processes at the two levels: (1) the evolution of macroscopic network properties, like diameter and network densification, via a series of network snapshots over time. (2) The network evolution at the level of individual edge arrivals and placements. Studying individual edge arrivals is important as it gives us clues to microscopic mechanisms that give rise to the observed macroscopic network properties. We study large online social networks with individual node and edge arrivals from the first to the “million-th” edge.

Observations: In both cases we make novel empirical observations. *E.g.*, the counterintuitive Densification power law and shrinking diameters at the macroscopic level, to link locality and triangle closure mechanisms taking place at the level of individual edges.

Models: We then use these observations to develop novel generation and evolution models that specify individual microscopic node behavior and give raise to the macroscopic phenomena observed in networks.

Algorithms: We also introduce a more mathematical model of Kronecker graphs, which is an analytically tractable network generation and evolution model. Moreover, we also present efficient algorithms to estimate Kronecker model parameters from data and then use them to generate synthetic graphs with similar properties as the original network.

Chapter 3

Macroscopic network evolution

How do real graphs evolve over time? What are “normal” growth patterns in social, technological, and information networks? Many studies have discovered patterns in *static graphs*, identifying properties in a single snapshot of a large network, or in a very small number of snapshots; these include heavy tails for in- and out-degree distributions, communities, small-world phenomena, and others. However, given the lack of information about network evolution over long periods, it has been hard to convert these findings into statements about trends over time.

Here we study a wide range of real graphs, and we observe some surprising phenomena. First, most of these graphs densify over time, with the number of edges growing super-linearly in the number of nodes. Second, the average distance between nodes often *shrinks* over time, in contrast to the conventional wisdom that such distance parameters should increase slowly as a function of the number of nodes (like $O(\log N)$ or $O(\log N / \log \log N)$, see Section 2.3).

Existing graph generation models do not exhibit these types of behavior, even at a qualitative level. We provide a new graph generator, based on a “forest fire” spreading process, that has a simple, intuitive justification, requires very few parameters (like the “flammability” of nodes), and produces graphs exhibiting the full range of properties observed both in prior work and in the present study.

3.1 Introduction

In recent years, there has been considerable interest in graph structures arising in technological, socio-logical, and scientific settings: computer networks (routers or autonomous systems connected together); networks of users exchanging e-mail or instant messages; citation networks and hyperlink networks; social networks (who-trusts-whom, who-talks-to-whom, and so forth); and countless more [Newman, 2003]. The study of such networks has proceeded along two related tracks: the measurement of large network datasets, and the development of random graph models that approximate the observed properties.

Many of the properties of interest in these studies are based on two fundamental parameters: the nodes’ *degrees* (*i.e.*, the number of edges incident to each node), and the *distances* between pairs of nodes (as measured by shortest-path length). The node-to-node distances are often studied in terms of the *diameter* — the maximum distance — and a set of closely related but more robust quantities including the average

distance among pairs and the *effective diameter* (the 90th percentile distance, a smoothed form of which we use for our studies).

Almost all large real-world networks evolve over time by the addition and deletion of nodes and edges. Most of the recent models of network evolution capture the growth process in a way that incorporates two pieces of “conventional wisdom.”

- (A) **Constant average degree assumption:** The average node degree in the network remains constant over time [Barabási and Albert, 1999, Kumar et al., 2000]. (Or equivalently, the number of edges grows linearly in the number of nodes.)
- (B) **Slowly growing diameter assumption:** The diameter is a slowly growing function of the network size, as in “small world” graphs [Albert et al., 1999, Broder et al., 2000, Milgram, 1967, Watts and Strogatz, 1998].

For example, the intensively-studied *preferential attachment model* [Barabási and Albert, 1999, Newman, 2003] posits a network in which each new node, when it arrives, attaches to the existing network by a constant number of out-links, according to a “rich-get-richer” rule. Recent work has given tight asymptotic bounds on the diameter of preferential attachment networks [Bollobás and Riordan, 2004, Chung and Lu, 2002a]; depending on the precise model, these bounds grow logarithmically [Krapivsky and Redner, 2005] or even slower than logarithmically in the number of nodes.

How are assumptions (A) and (B) reflected in data on network growth? Empirical studies of large networks to date have mainly focused on *static* graphs, identifying properties of a single snapshot or a very small number of snapshots of a large network. For example, despite the intense interest in the Web’s link structure, the recent work of Ntoulas et al. [Ntoulas et al., 2004] noted the lack of prior empirical research on the evolution of the Web. Thus, while one can assert based on these studies that, qualitatively, real networks have relatively small average node degrees and diameters, it has not been clear how to convert these into statements about trends over time.

The present work: Densification laws and shrinking diameters Here we study a range of different networks, from several domains, and we focus specifically on the way in which fundamental network properties vary with time. We find, based on the growth patterns of these networks, that principles (A) and (B) need to be reassessed. Specifically, we show the following for a broad range of networks across diverse domains.

- (A') **Empirical observation: Densification power laws:** The networks are becoming *denser* over time, with the average degree increasing (and hence with the number of edges growing super-linearly in the number of nodes). Moreover, the densification follows a power law pattern.
- (B') **Empirical observation: Shrinking diameters:** The effective diameter is, in many cases, actually *decreasing* as the network grows.

We view the second of these findings as particularly surprising: Rather than shedding light on the long-running debate over exactly how slowly the graph diameter *grows* as a function of the number of nodes, it suggests a need to revisit standard models so as to produce graphs in which the effective diameter is capable of actually *shrinking* over time. We also note that, while densification and decreasing diameters are properties that are intuitively consistent with one another (and are both borne out in the datasets we study), they are qualitatively distinct in the sense that it is possible to construct examples of graphs evolving over time that exhibit one of these properties but not the other.

We can further sharpen the quantitative aspects of these findings. In particular, the densification of these graphs, as suggested by (A') , is not arbitrary; we find that as the graphs evolve over time, they follow a version of the relation

$$E(t) \propto N(t)^a \quad (3.1)$$

where $E(t)$ and $N(t)$ denote the number of edges and nodes of the graph at time t , and a is an exponent that generally lies strictly between 1 and 2. We refer to such a relation as a *Densification Power Law (DPL)*. (Exponent $a = 1$ corresponds to constant average degree over time, while $a = 2$ corresponds to an extremely dense graph where each node has, on average, edges to a constant fraction of all nodes.)

What underlying process causes a graph to systematically densify, with a fixed exponent as in Equation (3.1), and to experience a decrease in effective diameter even as its size increases? This question motivates the second main contribution of this work: we present two families of probabilistic generative models for graphs that capture aspects of these properties. The first model, which we refer to as *Community Guided Attachment* (CGA) [Leskovec et al., 2005b], argues that graph densification can have a simple underlying basis; it is based on a decomposition of the nodes into a nested set of communities, such that the difficulty of forming links between communities increases with the community size. For this model, we obtain rigorous results showing that a natural tunable parameter in the model can lead to a densification power law with any desired exponent a . The second model, which is more sophisticated, exhibits both densification and a decreasing effective diameter as it grows. This model, which we refer to as the *Forest Fire Model*, is based on having new nodes attach to the network by “burning” through existing edges in epidemic fashion. The mathematical analysis of this model appears to lead to novel questions about random graphs that are quite complex, but through simulation we find that for a range of parameter values the model exhibits realistic behavior in densification, distances, and degree distributions. It is thus the first model, to our knowledge, that exhibits this full set of desired properties.

Accurate properties of network growth, together with models supporting them, have implications in several contexts.

- *Graph generation:* Our findings form means for assessing the quality of graph generators. Synthetic graphs are important for ‘what if’ scenarios, for extrapolations, and for simulations, when real graphs are impossible to collect (like, e.g., a very large friendship graph between people).
- *Graph sampling:* Datasets consisting of huge real-world graphs are increasingly available, with sizes ranging from the millions to billions of nodes. There are many known algorithms to compute interesting measures (shortest paths, centrality, betweenness, etc.), but most of these algorithms become impractical for large graphs. Thus sampling is essential — but sampling from a graph is a non-trivial problem since the goal is to maintain structural properties of the network. Densification laws can help discard bad sampling methods, by providing means to reject sampled subgraphs.

Our recent work [Leskovec and Faloutsos, 2006] proposed two views on sampling from large graphs. For *Back-in-time* sampling the goal is to find a sequence of sampled subgraphs that matches the evolution of the original graph and thus obey the temporal growth patterns. On the other hand, *Scale-down* sampling aims for a sample that matches the properties of the original large graph. We considered various sampling strategies, propose evaluation techniques, and use the temporal graph patterns presented in this chapter to evaluate the quality of the sampled subgraphs.

- *Extrapolations:* For several real graphs, we have a lot of snapshots of their past. What can we say about their future? Our results help form a basis for validating scenarios for graph evolution.

- *Abnormality detection and computer network management:* In many network settings, “normal” behavior will produce subgraphs that obey densification laws (with a predictable exponent) and other properties of network growth. If we detect activity producing structures that deviate significantly from this, we can flag it as an abnormality; this can potentially help with the detection of *e.g.* fraud, spam, or distributed denial of service (DDoS) attacks.

The rest of the chapter is organized as follows: Section 3.2 surveys the related work on networks over time. Section 3.3 gives our empirical findings on real-world networks across diverse domains. Section 3.4 describes our proposed models and gives results obtained both through analysis and simulation. Section 3.3.4 gives the formal and experimental analysis of the relationship between the degree distribution and the graph densification over time. We conclude and discuss the implications of our findings in Section 3.5.

3.2 Related work on graphs over time

Many network models are evolutionary in nature. For example, the *preferential attachment* [Abello et al., 2002, Barabási and Albert, 1999, Cooper and Frieze, 2003] is evolutionary as nodes arrive one at the time and each node creates its edges before next node arrives. Similar is true for *copying model* [Kleinberg et al., 1999, Kumar et al., 2000], which both produce graphs with constant average degree and logarithmically increasing diameter. A related *growing network with redirection* model [Krapivsky and Redner, 2001] produces networks with constant diameter and *logarithmically* increasing average degree over time [Krapivsky and Redner, 2005].

Similar to our Forest Fire Model is the work of Vazquez [Vazquez, 2001, 2003] where ideas based on random walks and recursive search for generating networks were introduced. In a random walk model the walk starts at a random node, follows links, and for each visited node with some probability an edge is created between the visited node and the new node. It can be shown that such model will generate graphs with power law degree distribution with exponent $\gamma \geq 2$. On the other hand, in the recursive search model first a new node is added to the network, and the edge to a random node is created. If an edge is created to a node in the network, then with some probability q an edge is also created to each of its 1-hop neighbors. This rule is recursively applied until no edges are created. The recursive search model is similar to our Forest Fire Model in a sense that it exploits current network structure to create new edges. However, there is an important difference that in recursive search model the average degree scales at most *logarithmically* (and not as a power law) with the number of nodes in the network. Our simulation experiments also indicated that the diameter of networks generated by the recursive search does not decrease over time, but it either slowly increases or remains constant.

It is important to note the fundamental contrast between one of our main findings here — that the average number of out-links per node is growing polynomially in the network size — and body of work on degree power laws. This earlier work developed models that almost exclusively used the assumption of node degrees that were bounded by constants (or at most logarithmic functions) as the network grew; our findings and associated model challenge this assumption, by showing that networks across a number of domains are becoming *denser* over time.

Dorogovtsev and Mendes in a series of works [Dorogovtsev and Mendes, 2001a,b, 2003] analyzed possible scenarios of nonlinearly growing networks while maintaining scale-free structure. Among considered hypothetical scenarios were also those where the number of links grows polynomially with the number

of edges, *i.e.*, Densification Power Law, while maintaining power law degree distribution. The authors call this an *accelerated growth* and propose preferential attachment type models where densification is forced by introducing an additional “node attractiveness” factor that is not only degree-dependent but also time-dependent. The motivation for their work comes from the fact that authors [Broder et al., 2000, Faloutsos et al., 1999] reported the increase of the average degree over time on the Web and the Internet. Our work differs in that it presents measurements on many time evolving networks to support our findings, and proposes generative models where densification is an emerging property of the model. Besides densification we also address the shrinking diameters and consider models for generating them.

The bulk of prior work on the empirical study of network datasets has focused on *static* graphs, identifying patterns in a single snapshot, or a small number of network snapshots (see also the discussion of this point by Ntoulas et al. [Ntoulas et al., 2004]). Two exceptions are the very recent work of Katz [Katz, 2005], who independently discovered densification power laws for citation networks, and the work of Redner [Redner, 2004], who studied the evolution of the citation graph of *Physical Review* over the past century. Katz’s work builds on his earlier research on power law relationships between the size and the recognition of professional communities [Katz, 1999]; his work on densification is focused specifically on citations, and he does not propose a generative network model to account for the densification phenomenon, as we do here. Redner’s work focuses on a range of citation patterns over time which are different from the network properties we study here.

Our Community Guided Attachment (CGA) model, which produces densifying graphs, is an example of a hierarchical graph generation model, in which the linkage probability between nodes decreases as a function of their relative distance in the hierarchy [Chakrabarti et al., 2004, Kleinberg, 2002, Watts et al., 2002, Leskovec et al., 2005b,a, Abello, 2004]. Again, there is a distinction between the aims of this past work and our model here; where these earlier network models were seeking to capture properties of individual snapshots of a graph, we seek to explain a time evolution process in which one of the fundamental parameters, the average node degree, is varying as the process unfolds. Our Forest Fire Model follows the overall framework of earlier graph models in which nodes arrive one at a time and link into the existing structure; like the copying model discussed above, for example, a new node creates links by consulting the links of existing nodes. However, the recursive process by which nodes in the Forest Fire Model creates these links is quite different, leading to the new properties discussed in the previous section.

3.3 Observations

We study the temporal evolution of several networks, by observing snapshots of these networks taken at regularly spaced points in time. We use datasets from seven different sources; for each, we have information about the time when each node was added to the network over a period of several years — this enables the construction of a snapshot at any desired point in time. For each of datasets, we find a version of the densification power law from Equation (3.1), $E(t) \propto N(t)^a$; the exponent a differs across datasets, but remains remarkably stable over time. We also find that the effective diameter decreases in all the datasets considered.

The datasets consist of two citation graphs for different areas in the physics literature, a citation graph for U.S. patents, a graph of the Internet, five bipartite affiliation graphs of authors with papers they authored, a recommendation network, an email communication network, and four online social networks. Overall, then, we consider 16 different datasets from 11 different sources.

SYMBOL	DESCRIPTION
G_t	Graph composed of nodes and edges that arrived before time t
N	Total number of nodes in a graph
E	Total number of edges in a graph
$N(t)$	Number of nodes in a graph at time t
$N(e)$	Number of nodes in a graph at time t
a	Power law densification exponent, $E(t) \propto N(t)^a$
c	<i>Difficulty Constant</i>
$f(h)$	<i>Difficulty Function</i>
b	Community hierarchy branching factor
\bar{d}	Expected average node out-degree in a graph
d_{max}	Maximum node out-degree in a graph
γ	Power law degree distribution exponent, $p(d) \propto d^{-\gamma}$
Γ	Community hierarchy (tree)
H_Γ	Height of the tree Γ
$h_\Gamma(v, w)$	Least common ancestor height of leaves v, w in Γ
$h(v, w)$	Length of the shortest path between nodes v, w
p	Forest Fire forward burning probability
p_b	Forest Fire backward burning probability
r	Ratio of backward and forward probability, $r = p/p_b$
α	Diameter factor (We fit $D^*(t) = \alpha \log t + \beta$ over time t). $\alpha > 0 \Rightarrow$ increasing, $\alpha < 0 \Rightarrow$ decreasing diameter

Table 3.1: Table of symbols.

3.3.1 Densification Laws

Here we describe the datasets we used, and our findings related to densification. For each graph dataset, we have, or can generate, several time snapshots, for which we study the number of nodes $N(t)$ and the number of edges $E(t)$ at each timestamp t . We denote by N and E the final number of nodes and edges. We use the term *Densification Power Law plot* (or just DPL plot) to refer to the log-log plot of number of edges $E(t)$ versus number of nodes $N(t)$.

ArXiv citation graph

We first investigate a citation graph provided as part of the 2003 KDD Cup [Gehrke et al., 2003]. The HEP-TH (high energy physics theory) citation graph from the e-print arXiv covers all the citations within a dataset of $N = 29,555$ papers with $E = 352,807$ edges. If a paper i cites paper j , the graph contains a directed edge from i to j . If a paper cites, or is cited by, a paper outside the dataset, the graph does not contain any information about this. We refer to this dataset as CIT-HEP-TH.

This data covers papers in the period from January 1993 to April 2003 (124 months). It begins within a few months of the inception of the arXiv, and thus represents essentially the complete history of its HEP-TH section. For each month m ($1 \leq m \leq 124$) we create a citation graph using all papers published up to month m . For each of these graphs, we plot the number of nodes versus the number of edges on a logarithmic scale and fit a line.

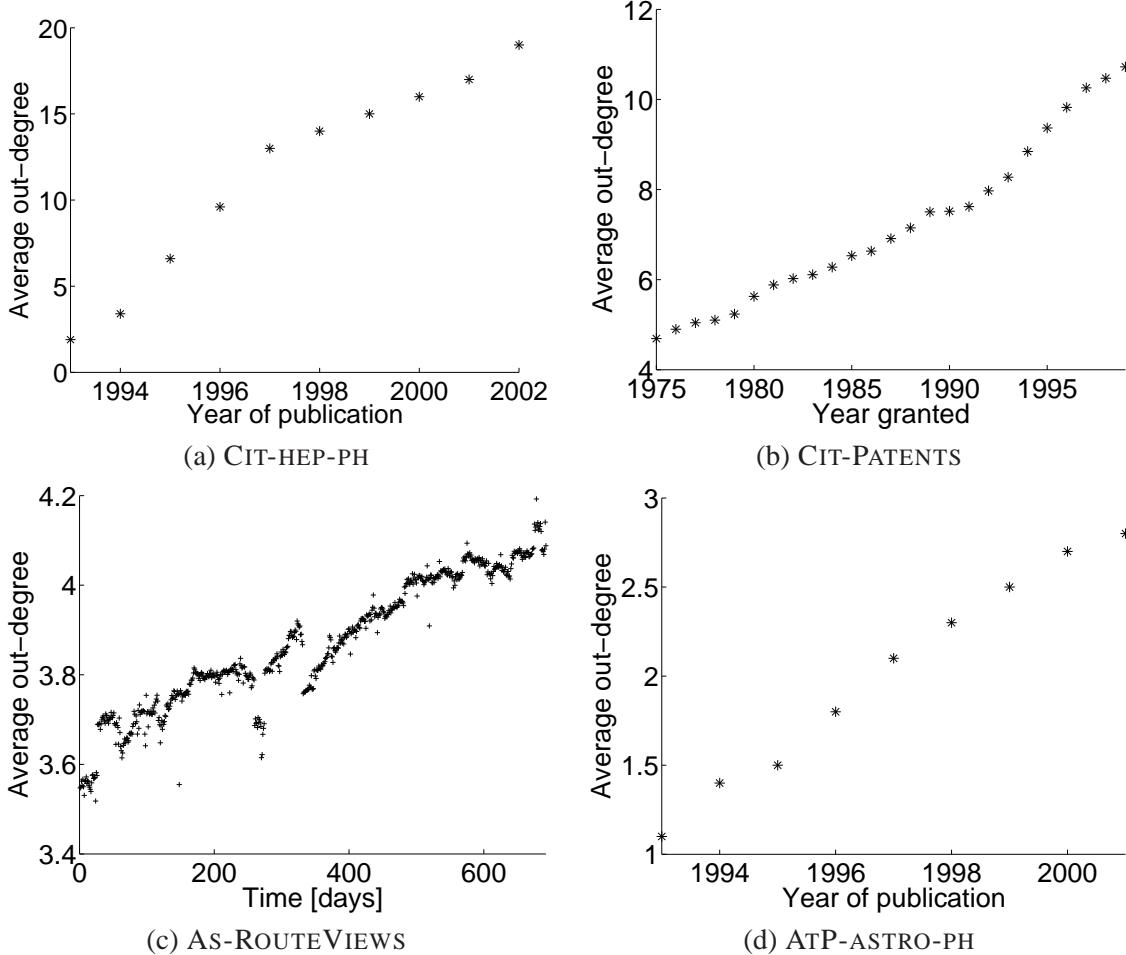


Figure 3.1: The average node out-degree over time for (a) ArXiv high energy physics citation network (CIT-HEP-TH), (b) US patent citation network (CIT-PATENTS), (c) Autonomous Systems network (AS-ROUTEVIEWS), (d) ArXiv Astro-Physics authors-to-papers bipartite network (ATP-ASTRO-PH). Notice that it increases, in all 4 datasets. That is, all graphs are *densifying*.

Figure 3.2(a) shows the DPL plot of the CIT-HEP-TH; the slope is $a = 1.68$ and corresponds to the exponent in the densification law. Notice that a is significantly higher than 1, indicating a large deviation from linear growth. As noted earlier, when a graph has $a > 1$, its average degree increases over time. Figure 3.1(a) exactly plots the average degree \bar{d} over time, and it is clear that \bar{d} increases. This means that the average length of the bibliographies of papers increases over time. We also found that the median of the degree distribution over time also behaves in a qualitatively similar way, *i.e.*, it increases over time.

There is a subtle point here that we elaborate next: With almost any network dataset, one does not have data reaching all the way back to the network's birth (to the extent that this is a well-defined notion). We refer to this as the problem of the “*missing past*.” Due to this, there will be some effect of increasing out-degree simply because edges will point to nodes prior to the beginning of the observation period, *i.e.*, over time less references are pointing to papers outside the dataset. We refer to such nodes as *phantom nodes*, with a similar definition for *phantom edges*. In all our datasets, we find that this effect is relatively minor once we move away from the beginning of the observation period; on the other hand, the phenomenon

of increasing degree continues through to the present. For example, in arXiv, nodes over the most recent years are primarily referencing non-phantom nodes; we observe a knee in Figure 3.1(a) in 1997 that appears to be attributable in large part to the effect of phantom nodes. (Later, when we consider a graph of the Internet and the online social networks, we will see a case where comparable properties hold in the absence of any “missing past” issues.) A similar observation of growing reference lists over time was also independently made by Krapivsky and Redner [Krapivsky and Redner, 2005].

We also experimented with a second citation graph CIT-HEP-PH, taken from the HEP-PH section of the arXiv, which is about the same size as our first arXiv dataset. It exhibits the same behavior, with the densification exponent $a = 1.56$. The plot is omitted but we show the summary of results on all 16 datasets we considered in table 3.2.

Patents citation graph

Next, we consider a U.S. patent citation dataset maintained by the National Bureau of Economic Research [Hall et al., 2001]. The data set spans 37 years (January 1, 1963 to December 30, 1999), and includes all the utility patents granted during that period, totaling $N = 3,923,922$ patents. The citation graph includes all citations made by patents granted between 1975 and 1999, totaling $E = 16,522,438$ citations. For the patents dataset there are 1,803,511 nodes for which we have no information about their citations (we only have the in-links). Because the dataset begins in 1975, it too has a “missing past” issue, but again the effect of this is minor as one moves away from the first few years. We refer to this patent citation network as CIT-PATENTS.

The CIT-PATENTS patents data also contains citations outside the dataset. For patents outside the dataset the time is unknown. These patents have zero out-degree and are at some time cited by the patents from within the dataset. We set the time (grant year) of these out-of-dataset patents to the year when they were first cited by a patent from the dataset. This is natural and is equivalent to saying that patents for which grant year is unknown are in the dataset from the beginning, but when counting, we count only non-zero degree nodes. So the time when we first count an unknown patent is when it gets a first link.

We follow the same procedure as with arXiv citation networks. For each year Y from 1975 to 1999, we create a citation network on patents up to year Y , and give the DPL plot, in Figure 3.2(b). As with the arXiv citation network, we observe a high densification exponent, in this case $a = 1.66$.

Figure 3.1(b) illustrates the increasing out-degree of patents over time. Note that this plot does not incur any of the complications of a bounded observation period, since the patents in the dataset include complete citation lists, and here we are simply plotting the average size of these as a function of the year.

Autonomous systems graph

The graph of routers comprising the Internet can be organized into sub-graphs called Autonomous Systems (AS). Each AS exchanges traffic flows with some neighbors (peers). We can construct a communication network of who-talks-to-whom from the BGP (Border Gateway Protocol) logs.

We use the *Autonomous Systems (AS)* dataset from RouteViews project at University of Oregon [RouteViews, 1997]. The dataset contains 735 daily instances which span an interval of 785 days from November 8 1997 to January 2 2000. The graphs range in size from $N = 3,011$ nodes and $E = 10,687$ edges to

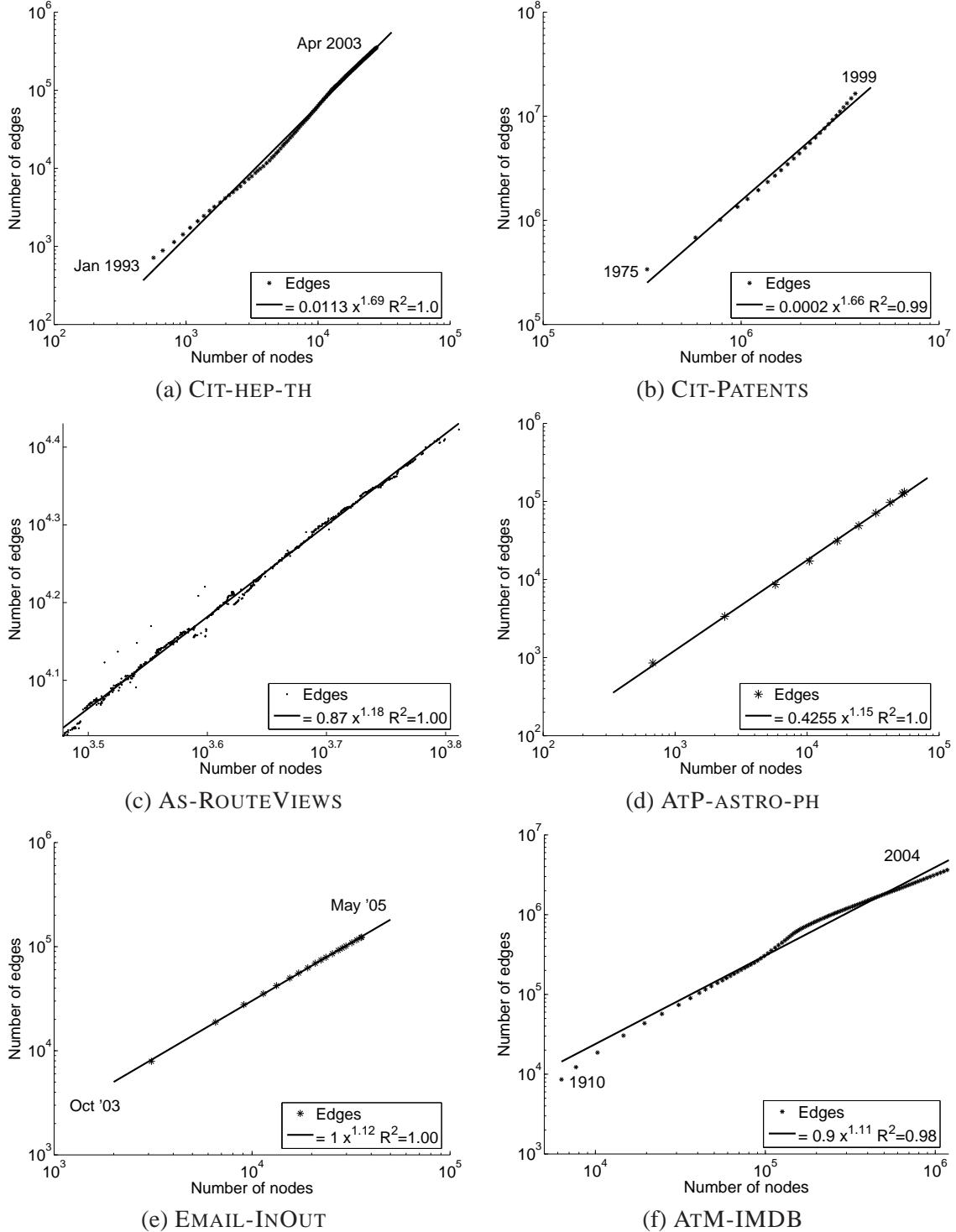


Figure 3.2: Number of edges $E(t)$ versus number of nodes $N(t)$, in log-log scales, for (a) ArXiv high energy physics citation network (CIT-HEP-TH), (b) US patent citation network (CIT-PATENTS), (c) Autonomous Systems network (AS-ROUTEVIEWS), (d) ArXiv Astro-Physics authors-to-papers bipartite network (ATP-ASTRO-PH), (e) Email network (EMAIL-INOUT), and (f) Actors-to-movies bipartite network from IMDB (ATM-IMDB). All 6 graphs obey the Denification Power Law, with a consistently good fit. Slopes: $a = 1.68, 1.66, 1.18, 1.15, 1.12$, and 1.11 respectively.

the largest AS graph that has $N = 6,474$ nodes and $E = 26,467$ edges. We refer to this dataset as AS-ROUTEVIEWS.

In contrast to citation networks, where nodes and edges only get added (not deleted) over time, the AS dataset also exhibits both the addition and deletion of the nodes and edges over time.

Figure 3.2(c) shows the DPL plot for the AS-ROUTEVIEWS dataset. We observe a clear trend: Even in the presence of noise, changing external conditions, and disruptions to the Internet we observe a strong super-linear growth in the number of edges over more than 700 AS graphs. We show the increase in the average node degree over time in Figure 3.1(c). The densification exponent is $a = 1.18$, lower than the one for the citation networks, but still clearly greater than 1.

Affiliation graphs

Using the arXiv data, we also constructed bipartite *affiliation graphs*. There is a node for each paper, a node for each person who authored at least one arXiv paper, and an edge connecting people to the papers they authored. Note that the more traditional *co-authorship network* is implicit in the affiliation network: two people are co-authors if there is at least one paper joined by an edge to each of them.

We studied affiliation networks derived from the five largest categories in the arXiv. We refer to these Authors-to-Papers graphs as ATP-ASTRO-PH, ATP-HEP-TH, ATP-HEP-PH, ATP-COND-MAT and ATP-GR-QC. See also the table A.3 for additional information about the datasets.

We place a time-stamp on each node: the submission date of each paper, and for each person, the date of their first submission to the arXiv. The data for affiliation graphs covers the period from April 1992 to March 2002. The smallest of the graphs (category GR-QC) had 19,309 nodes (5,855 authors, 13,454 papers) and 26,169 edges. ATP-ASTRO-PH is the largest graph, with 57,381 nodes (19,393 authors, 37,988 papers) and 133,170 edges. It has 6.87 authors per paper; most of the other categories also have similarly high numbers of authors per paper.

For all these affiliation graphs we observe similar phenomena, and in particular we have densification exponents between 1.08 and 1.15. We present the complete set of measurements only for ASTRO-PH, the largest affiliation graph. Figures 3.1(d) and 3.2(d) show the increasing average degree over time, and a densification exponent of $a = 1.15$. Table 3.2 shows the sizes and Densification Power Law exponents for other four affiliation graphs.

Email network

We also considered an email network from a large European research organization. For a period from October 2003 to May 2005 (18 months) we have anonymized information about all incoming and outgoing email of the research organization. For each sent or received email message we know the time, the sender and the recipient of the email. All personally identifiable data was hashed and nodes were assigned random ids. Overall we have 3,038,531 emails between 287,755 different email addresses. Note that we have a complete email graph for only 1,258 email addresses that come from inside the research organization. Furthermore, there are 35,756 email addresses that both sent and received email within the span of our dataset. All other email addresses are either non-existing, mistyped or spam.

Given a set of email messages we need to create a graph. Since there can be multiple emails sent between same two addresses (nodes) we follow the practice of Kossinets and Watts [Kossinets and Watts, 2006]. Given a set of email messages, each node corresponds to an email address. We create an edge between nodes i and j , if they exchanged messages both ways, *i.e.*, node i sent at least one message to node j , and j sent at least one message to i .

Similarly to citation networks, we take all email messages up to particular time t and create a graph using the procedure described above. So, in the first month we observe 254,080 emails between 38,090 different addresses. Using the procedure [Kossinets and Watts, 2006] of generating a graph from a set of emails, we get $N = 6,537$ nodes and $E = 18,812$ edges. After 18 months, at the end of the dataset, we have $N = 35,756$ nodes and $E = 123,254$ edges. We refer to this network as EMAIL-INOUT. See also table A.2 for additional information about the dataset.

Figure 3.2(e) presents the DPL plot for the EMAIL-INOUT network. Observe a clear trend: the email network is densifying, regardless of the fact that it is growing and that new parts of social network (email address space) are being explored. The densification exponent is $a = 1.12$, lower than the one for the citation networks but more similar to those from affiliation networks. Still clearly greater than 1.

Note that there is one issue with this dataset: we have complete information about all sent and received emails only for the core of the network (1258 email addresses from the organization). For the rest of the addresses, the nodes on the periphery, we only have their communication (links) with the core of the network.

Regardless of how we look at the email network it always densifies: If we consider only the core of the network, the densification is very high. This is expected, since the number of nodes (people at the research organization) basically remains constant over time and the edges can only be added, not deleted, and densification naturally occurs.

The EMAIL-INOUT network also densifies if we consider the core plus the periphery but when determining edges we take a 2 month sliding window [Kossinets and Watts, 2006]. This means that for every month m , we take all email messages between $m - 1$ and m , and create a graph, where there is an edge, if nodes exchanged emails both ways in the last 2 months. This graph also densifies with densification exponent $a = 1.21$.

Interestingly, the sliding window email network has higher densification exponent than the full evolving email network. A possible explanation is that email usage is increasing over time and not all nodes (email addresses) are active at all times. Over the 18 month time period the size of 2-month sliding window graphs increases from 7,000 to 10,000 nodes. On the other hand the full email graph (composed of all nodes up to month m) grows from 3,000 to 38,000 nodes over the same time period. This means that there is a large number of e-mail addresses that are active only for a period of time. In a moving window graph we observe only active users and thus more edges since email usage has also increased and people communicate more. As opposed to the evolution of the full email network, the moving window graphs do not have to accumulate the history, *i.e.*, sparse graphs from the past, so they densify faster.

IMDB actors to movies network

The Internet Movie Data Base (IMDB, <http://www.imdb.com>) is a collection of facts about movies and actors. For every movie we know the year of production, genre, and actor names that appeared in the

movie. From IMDB we obtained data about 896,192 actors and 334,084 movies produced between 1890 and 2004 (114 years).

Given this data we created a bi-partite graph of actors to movies the same way as in the case of affiliation networks. We refer to this network as ATM-IMDB. This means that whenever a new movie appears, it links to all the actors participating in it. We create a new actor node when the actor first appears in any movie. This way, when a new movie appears, we first create a movie node. Then we introduce actor nodes, but only for actors for whom this was their first appearance in a movie. Then we link actors and the movie.

In our experiment we started observing the graph in 1910, when the giant connected component started to form. Before 1910 the largest connected component consisted of less than 15% of the nodes. At the beginning of our observation period the ATM-IMDB network had $N = 7,690$ nodes (4,219 actors and 3,471 movies) and $E = 12,243$ edges. At the end of the dataset in 2004, we have $N = 1,230,276$ nodes and $E = 3,790,667$ edges. See also table A.4 for additional information about the dataset.

We follow the usual procedure: for every year Y we take all the movies up to year Y and actors that appeared in them. We create a graph and measure how the number of edges grows with the number of nodes. Figure 3.2(f) presents the DPL plot for the ATM-IMDB actors to movies network. Again, notice the nontrivial densification exponent of $a = 1.11$.

Product recommendation network

We also report the analysis of the product recommendation network [Leskovec et al., 2006a] that we will describe in greater detail in chapter 6. We measure the densification of a large person-to-person recommendation network from a large on-line retailer. Nodes represent people and edges represent recommendations. The network generation process was as follows. Each time a person *purchases* a book, music CD, or a movie he or she is given the option of sending emails recommending the item to friends. Any of the recipients of the recommendation that makes a purchase can further recommend the item, and by this propagation of recommendations the network forms. We refer to this network as RECOMMENDATIONS.

The RECOMMENDATIONS network consists of $E = 15,646,121$ recommendations made among $N = 3,943,084$ distinct users. The data was collected from June 5 2001 to May 16 2003. In total, 548,523 products were recommended. We report the Densification Power Law exponent $a = 1.26$ in table 3.2.

Online social networks

We also consider large online social networks that are parts of the popular social networking and photo sharing websites like: FLICKR (flickr.com, a photo-sharing website), DELICIOUS (del.icio.us, a collaborative bookmark tagging website), YAHOO! ANSWERS (answers.yahoo.com, a knowledge sharing website), and LINKEDIN (linkedin.com, a professional contacts website) — where nodes represent people and edges represent social relationships. Networks have up to 8 million nodes and 31 million edges. Notice here we have complete data on the evolution of these four networks from the inception of the service to the end. All personally identifiable data was hashed and nodes were assigned random ids. Refer to table 3.2 for information on network sizes and densification exponents.

DATASET	NODES	EDGES	TIME	DPL EXPONENT
CIT-HEP-PH	30,501	347,268	10 years	1.56
CIT-HEP-TH	29,555	352,807	10 years	1.68
CIT-PATENTS	3,923,922	16,522,438	37 years	1.66
AS-ROUTEVIEWS	6,474	26,467	785 days	1.18
ATP-ASTRO-PH	57,381	133,179	10 years	1.15
ATP-COND-MAT	62,085	108,182	10 years	1.10
ATP-GR-QC	19,309	26,169	10 years	1.08
ATP-HEP-PH	51,037	89,163	10 years	1.08
ATP-HEP-TH	45,280	68,695	10 years	1.08
EMAIL-INOUT	35,756	123,254	18 months	1.12
ATM-IMDB	1,230,276	3,790,667	114 years	1.11
RECOMMENDATIONS	3,943,084	15,656,121	710 days	1.26
FLICKR	584,207	3,554,130	20 months	1.32
DELICIOUS	203,234	430,707	10 months	1.15
ANSWERS	598,314	1,834,217	4 months	1.25
LINKEDIN	7,550,955	30,682,028	3.5 years	1.20

Table 3.2: Dataset names with sizes, time lengths and Densification Power Law exponents. Notice very high densification exponent for citation networks (≈ 1.6), around 1.2 for Autonomous Systems and lower (but still significant) densification exponent (≈ 1.1) for affiliation and collaboration type networks.

3.3.2 Shrinking Diameters

We now discuss the behavior of the effective diameter over time, for this collection of network datasets. Following the conventional wisdom on this topic, we expected the underlying question to be whether we could detect the differences among competing hypotheses concerning the growth rates of the diameter — for example, the difference between logarithmic and sub-logarithmic growth. Thus, it was with some surprise that we found the effective diameters to be actually *decreasing* over time (Figure 3.3).

As mentioned earlier in Chapter 2, a graph has diameter D if every pair of nodes can be connected by a path of length at most D . The diameter D is susceptible to outliers. Thus, a more robust measure of the pairwise distances between nodes of a graph is the *effective diameter*. This is defined as the minimum number of hops in which 90% of all connected pairs of nodes can reach each other. See section 2.1 for more precise definitions of these concepts. The effective diameter has been found to be small for large real-world graphs, like Internet, Web, and social networks [Albert and Barabási, 2002, Milgram, 1967].

We follow the same procedure as in case of Densification Power Law measurements. For each time t , we create a graph consisting of nodes up to that time, and compute the effective diameter of the undirected version of the graph.

Figure 3.3 shows the effective diameter over time; one observes a decreasing trend for all the graphs. We performed a comparable analysis to what we describe here for all 16 graph datasets in our study, with very similar results. For the citation networks in our study, the decreasing effective diameter has the following interpretation: Since all the links out of a node are “frozen” at the moment it joins the graph, the decreasing distance between pairs of nodes appears to be the result of subsequent papers acting as “bridges” by citing

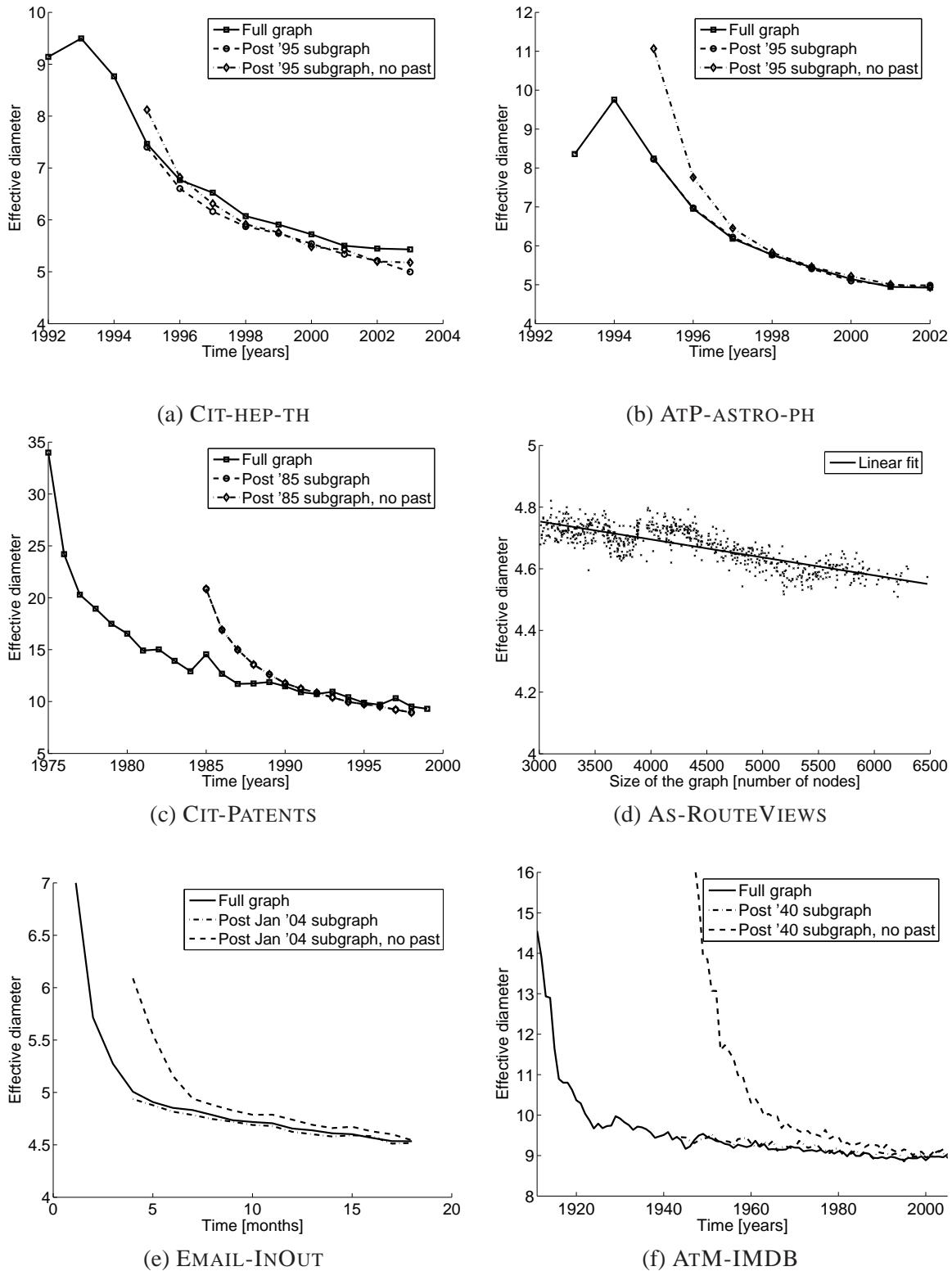


Figure 3.3: The effective diameter over time for 6 different datasets. Notice consistent decrease of the diameter over time.

earlier papers from disparate areas. Note that for other graphs in our study, such as the AS dataset, it is possible for an edge between two nodes to appear at an arbitrary time after these two nodes join the graph.

We note that the effective diameter of a graph over time is necessarily bounded from below, and the decreasing patterns of the effective diameter in the plots of Figure 3.3 are consistent with convergence to some asymptotic value. However, understanding the full “limiting behavior” of the effective diameter over time, to the extent that this is even a well-defined notion, remains an open question.

Validating the shrinking diameter conclusion

Given the unexpected nature of this result, we wanted to verify that the shrinking diameters were not attributable to artifacts of our datasets or analyses. We explored this issue in a number of ways, which we now summarize; the conclusion is that the shrinking diameter appears to be a robust, and intrinsic, phenomenon. Specifically, we performed experiments to account for (a) possible sampling problems, (b) the effect of disconnected components, (c) the effect of the “missing past”(as in the previous subsection), and (d) the dynamics of the emergence of the giant component.

- *Possible sampling problems:* Computing shortest paths among all node pairs is computationally prohibitive for graphs of our scale. We used several different approximate methods, obtaining almost identical results from all of them. In particular, we applied the Approximate Neighborhood Function (ANF) approach [Palmer et al., 2002] (in two different implementations), which can estimate effective diameters for very large graphs, as well as a basic sampling approach in which we ran exhaustive breadth-first search from a subset of the nodes chosen uniformly at random. The results using all these methods were essentially identical.

Plots on figure 3.3 were created by averaging over 100 runs of the ANF, the approximate diameter algorithm. For all datasets the standard error is less than 10%. For clarity of presentation we do not show the error bars.

- *Disconnected components:* One can also ask about the effect of small disconnected components. All of our graphs have a single *giant component* – a connected component (or a weakly connected component in the case of directed graphs, ignoring the direction of the edges) that accounts for a significant fraction of all nodes. For each graph, we computed effective diameters for both the entire graph and for just the giant component; again, our results are essentially the same using these two methods.
- *“Missing Past” effects:* A third issue is the problem of the “missing past,” the same general issue that surfaced in the previous subsection when we considered densification. In particular, we must decide how to handle citations to papers that predate our earliest recorded time. (Note that the missing past is not an issue for the AS network and the four online social network data, where the effective diameter also decreases.)

To understand how the diameters of our networks are affected by this unavoidable problem, we perform the following test. We pick some positive time $t_0 > 0$, and determine what the diameter would look like as a function of time, *if this were the beginning of our data*. We then put back in the nodes and the edges from before time t_0 , and study how much the diameters change. If this change is small — or at least if it does not affect the qualitative conclusions — then it provides evidence that the missing past is not influencing the overall result.

Specifically, we set this cut-off time t_0 to be the beginning of 1995 for the arXiv (since we have data from 1993), and to be 1985 for the patent citation graph (we have data from 1975). For Email network we set the cut-off time to January 2004 and for IMDB to 1940 (we also experimented with 1920 and 1960 and findings were consistent). We then compared the results of three measurements:

- *Diameter of full graph.* For each time t we compute the effective diameter of the graph’s giant component.
- *Post- t_0 subgraph.* We compute the effective diameter of the post- t_0 subgraph using all nodes and edges. This means that for each time t ($t > t_0$) we create a graph using all nodes dated before t . We then compute the effective diameter of the subgraph of the nodes dated between t_0 and t . To compute the effective diameter we can use all edges and nodes (including those dated before t_0). This means that we are measuring distances *only* among nodes dated between t_0 and t while also using nodes and edges before t_0 as “shortcuts” or “bypasses”. The experiment measures the diameter of the graph if we knew the full (pre- t_0) past — the citations of the papers which we have intentionally excluded for this test.
- *Post- t_0 subgraph, no past.* We set t_0 the same way as in previous experiment, but then for all nodes dated before t_0 we delete all their out-links. This creates the graph we would have gotten if we had started collecting data only at time t_0 .

In Figure 3.3, we superimpose the effective diameters using the three different techniques. If the missing past does not play a large role in the diameter, then all three curves should lie close to one another. We observe this is the case for the arXiv citation graphs. For the arXiv paper-author affiliation graph, and for the patent citation graph, the curves are quite different right at the cut-off time t_0 (where the effect of deleted edges is most pronounced), but they quickly align with one another. As a result, it seems clear that the continued decreasing trend in the effective diameter as time runs to the present is not the result of these boundary effects.

- *Emergence of giant component:* A final issue is the dynamics by which the giant component emerges. For example, in the standard Erdős–Rényi random graph model (which has a substantially different flavor from the growth dynamics of the graphs here), the diameter of the giant component is quite large when it first appears, and then it shrinks as edges continue to be added. Could shrinking diameters in some way be a symptom of emergence of giant component?

It appears fairly clear that this is not the case. Figure 3.4 shows the fraction of all nodes that are part of the largest connected component (LCC) over time. We plot the size of the LCC for the full graph and for a graph where we had no past — *i.e.*, where we delete all out-links of the nodes dated before the cut-off time t_0 . Because we delete the out-links of the pre- t_0 nodes the size of LCC is smaller, but as the graph grows the effect of these deleted links becomes negligible.

We see that within a few years the giant component accounts for almost all the nodes in the graph. The effective diameter, however, continues to steadily decrease beyond this point. This indicates that the decrease is happening in a “mature” graph, and not because many small disconnected components are being rapidly glued together.

Based on all this, we believe that the decreasing diameters in our study reflect a fundamental property of the underlying networks. Understanding the possible causes of this property, as well as the causes of the densification power laws discussed earlier, will be the subject of the next section.

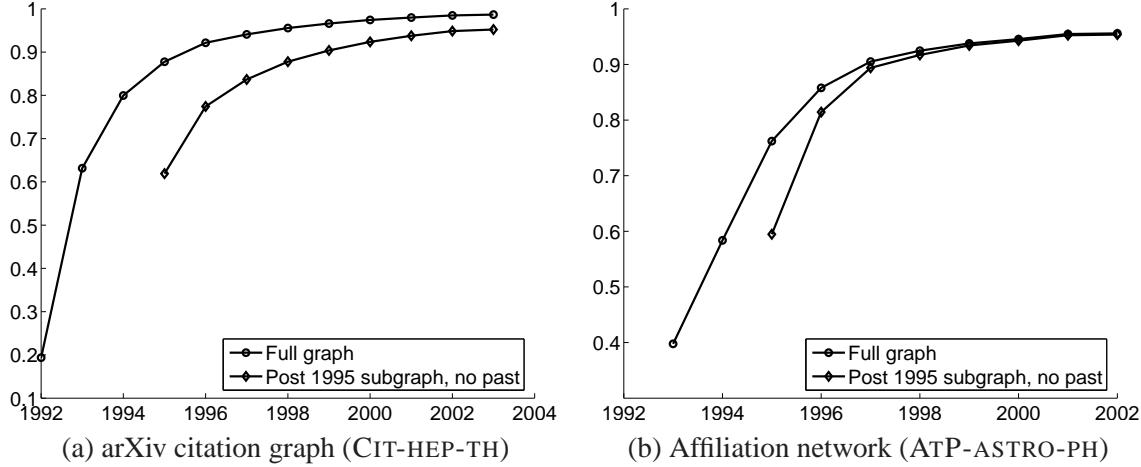


Figure 3.4: The fraction of nodes that are part of the giant connected component over time. We see that after 4 years the 90% of all nodes in the graph belong to giant component.

3.3.3 Does densification cause shrinking diameter?

A natural question to ask next is whether densification itself is enough for the diameter to shrink. Or, is there something more that causes shrinking diameters. For example, it could be that the edge attachment changes and the edges attach less and less “locally” over time, which shrinks the network diameter.

In principle there are three possible answers to this question. We list them from the simplest to the most complex: (1) densification causes shrinking diameter; (2) densification in combination with particularly evolving degree sequence causes shrinking diameter; (3) densification and special evolution of edge attachment cause shrinking diameter. Next, we examine which of these possible answers is true.

First, we examine the connection between the densification and the shrinking diameter. We generate a densifying random graph $G_{n,p}$ and measure the effective diameter as we grow and densify the graph. If solely the densification causes shrinking diameter, then the diameter of a densifying $G_{n,p}$ should also shrink. Figure 3.5(a) shows the plot for a densifying random graph with densification exponent $a = 1.3$. Notice the diameter is still slowly increasing which shows that densification itself is not enough to obtain shrinking diameter. Similarly, Figure 3.5(b) shows the diameter of a densifying Preferential Attachment (PA) model. Here the diameter quickly fluctuates and then remains constant with the network size.

Now, we evaluate the hypothesis whether the densification and the evolution of the degree sequence could cause the diameter to shrink. We measure the diameter over time of a real network and then compare this with a diameter of a random network conditioning on the same degree distribution. Basically, we take a real network and then generate a random network with same degree sequence using the configuration model [Bollobas, 1980]. Figures 3.5(c) and (d) show the true network shrinking diameter for the ATP-ASTRO-PH affiliation network the US patent citation network (CIT-PATENTS). Dots present the diameter of the real network, while line shows the evolution of the diameter of a “rewired” network, *i.e.*, a random network with same degree distribution. Notice the effective diameter nicely follows true diameter even if we randomly rewire the edges. This shows that there is nothing special about how the edges attach but it is the way the degree sequence evolves over time that gives rise to the shrinking diameter.

Next, we analyze exactly the connections between the densification power law and the evolution of the degree sequence.

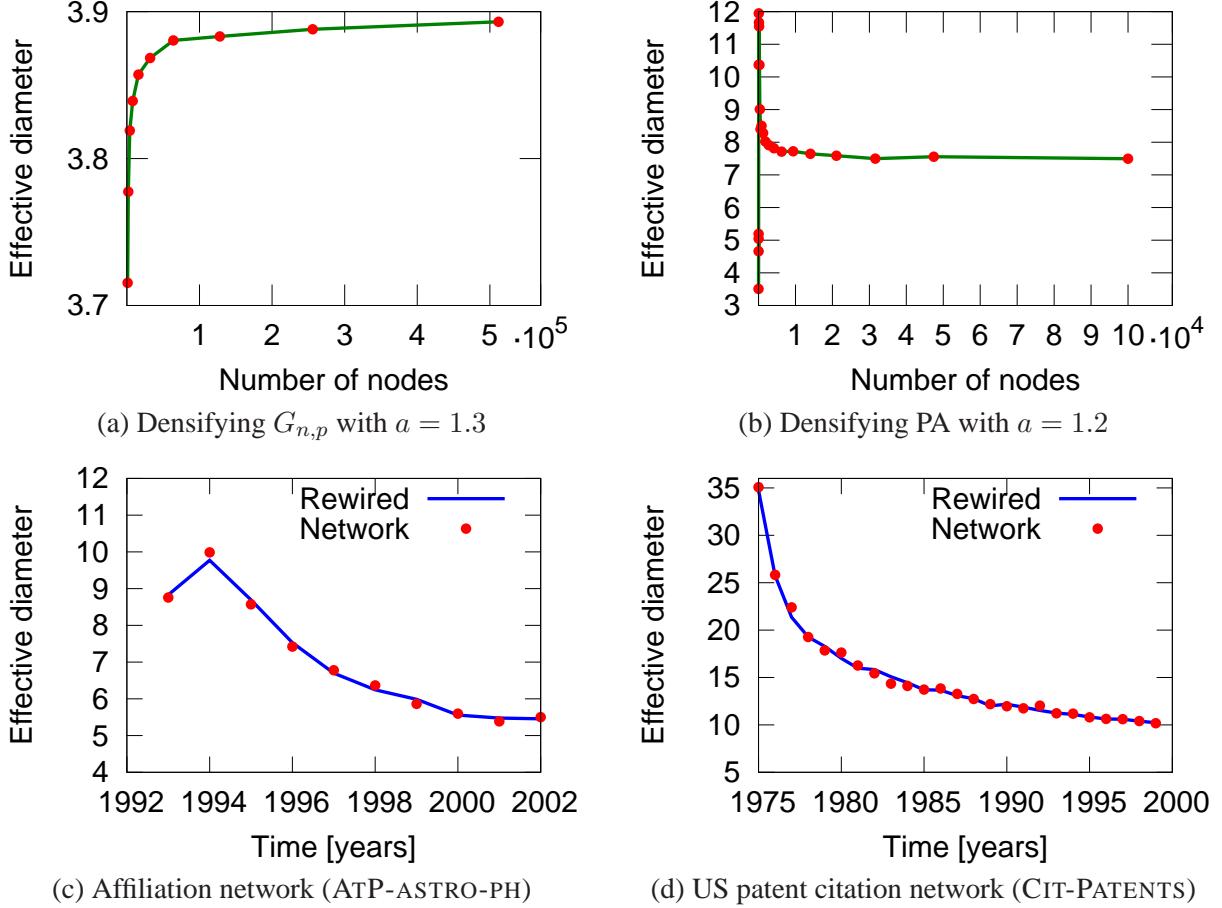


Figure 3.5: (Top row: (a) The effective diameter of a densifying Erdős–Rényi random graph $G_{n,p}$ with densification exponent $a = 1.3$. (b) Densifying Preferential Attachment (PA) model with densification exponent $a = 1.2$. In $G_{n,p}$ the diameter is still slowly increasing regardless of the fact that the network is densifying. In case of Preferential Attachment the diameter is basically constant. This means that densification itself is not enough for diameter to shrink. Bottom row compares the true effective diameter (red dots) with the effective diameter of a rewired network, *i.e.*, a random graph with same degree distribution (solid line). Notice they both match well. The rewiring process does not alter network’s degree sequence and densification. This shows one needs the right combination of the densification and the evolving degree sequence to obtain shrinking diameter.

3.3.4 Densification and the degree distribution over time

Many real world graphs exhibit power law degree distributions [Barabási and Albert, 1999, Faloutsos et al., 1999]. As we saw in section 3.3 the average degree increases over time, and the graphs densify following the power law relationship between the number of nodes and the number of edges. Here we analyze the relation between the densification and the power law degree distribution over time, and find evidence that some of the real world graphs obey the relations we find. A similar analysis was also performed by Dorogovtsev and Mendes [Dorogovtsev and Mendes, 2002b] although without specific measurements or comparison to real data.

We analyze the following two cases: If the degree distribution of a time evolving graph is power law, and it maintains *constant* power law exponent γ over time, then we show that for $1 < \gamma < 2$ we obtain the Densification Power Law exponent

$$a = 2/\gamma.$$

arises. In this case the Densification Power Law is the consequence of the fact that a power law distribution with exponent $\gamma < 2$ has no finite expectation [Newman, 2005], and thus the average degree grows with the number of samples (*i.e.*, nodes) while power law degree exponent is constant over time.

Our second result is for the case when temporally evolving graph densifies with densification exponent a , and follows a power law degree distribution with exponent $\gamma > 2$ that we allow to *change* over time. We show that in this case for a given densification exponent a , the power law degree exponent $\gamma(N)$ has to evolve with the size of the graph N as

$$\gamma(N) = \frac{4N^{a-1} - 1}{2N^{a-1} - 1}$$

This shows that Densification Power Law and the degree distribution are related and that one implies the other.

Constant degree exponent over time

First, we analyze the case where the graph over time maintains power law degree distribution with a constant exponent γ . Power law distribution $p(x) = cx^{-\gamma}$ with exponent $\gamma < 2$ has infinite expectation [Newman, 2005], *i.e.*, as the number of samples increases, the average also increases. Assuming that the exponent (slope) of the degree distribution does *not change over time*, a natural question to ask is: *what is the relation between the Densification Power Law exponent and the degree distribution over time?* The following theorem answers the question:

Theorem 3.3.1. *In a temporally evolving graph with a power law degree distribution having constant degree exponent γ over time, the DPL exponent a is:*

$$a = 1 \quad \text{if } \gamma > 2 \tag{3.2}$$

$$= 2/\gamma \quad \text{if } 1 \leq \gamma \leq 2 \tag{3.3}$$

$$= 2 \quad \text{if } \gamma < 1 \tag{3.4}$$

Proof. Assume that at any time t the degree distribution of an undirected graph G follows a power law. This means the number of nodes N_d with degree d is $N_d = cd^{-\gamma}$, where c is a constant. Now assume that at some point in time the maximum degree in the graph is d_{max} . Later as the graph grows we will let $d_{max} \rightarrow \infty$. Using the previous power law relation, we can calculate the number of nodes N and the number of edges E in the graph:

$$\begin{aligned} N &= \sum_{d=1}^{d_{max}} cd^{-\gamma} \approx \int_{d=1}^{d_{max}} d^{-\gamma} = c \frac{d_{max}^{1-\gamma} - 1}{1 - \gamma} \\ E &= \frac{1}{2} \sum_{d=1}^{d_{max}} cd^{1-\gamma} \approx \int_{d=1}^{d_{max}} d^{1-\gamma} = c \frac{d_{max}^{2-\gamma} - 1}{2 - \gamma} \end{aligned}$$

Now, we let the graph grow, so $d_{max} \rightarrow \infty$. Then the DPL exponent a is:

$$a = \lim_{d_{max} \rightarrow \infty} \frac{\log(E)}{\log(N)} = \frac{\gamma \log(d_{max}) + \log(|d_{max}^{2-\gamma} - 1|) - \log(|2 - \gamma|)}{\gamma \log(d_{max}) + \log(|d_{max}^{1-\gamma} - 1|) - \log(|1 - \gamma|)}$$

Note, that the degree distribution exponent is γ , so we also have the relation $\log(c) = \gamma \log(d_{max})$. Now, we have 3 cases:

Case 1: $\gamma \geq 2$. No densification:

$$a = \frac{\gamma \log(d_{max}) + o(1)}{\gamma \log(d_{max}) + o(1)} = 1$$

Case 2: $1 < \gamma < 2$ is the interesting case where densification arises:

$$a = \frac{\gamma \log(d_{max}) + (2 - \gamma) \log(d_{max}) + o(1)}{\gamma \log(d_{max}) + o(1)} = \frac{2}{\gamma}$$

Case 3: $\gamma \leq 1$. Maximum densification – the graph is basically a clique and the number of edges grows quadratically with the number of nodes:

$$a = \frac{\gamma \log(d_{max}) + (2 - \gamma) \log(d_{max}) + o(1)}{\gamma \log(d_{max}) + (1 - \gamma) \log(d_{max}) + o(1)} = 2$$

□

This shows that for cases when graph evolves by maintaining the constant power law degree exponent $\gamma > 2$ over time it does not densify. However, for cases when $\gamma < 2$ we observe densification. This can easily be explained. The densification means that the number of edges grows faster than the number of nodes. So, for densification to appear the tail of the degree distribution has to grow, *i.e.* has to accumulate more mass over time. Here, this is the case since power law distributions with exponent $\gamma < 2$ have no finite expectation. In the case of degree distribution this means that the expected node degree grows as the graph accumulates more nodes (*i.e.*, samples from degree distribution).

Evolving degree distribution

There also exist graphs with degree distribution $\gamma > 2$ which can also densify. Now, we allow the degree distribution to change over time. In fact, the degree distribution has to flatten over time to accumulate more mass in the tail as more nodes are added to allow for densification. This is what we explore next.

In the previous section we assumed that the exponent γ of the power law degree distribution remains constant over time, and then found the range for power law degree exponent γ where it leads to densification. Now, we assume Densification Power Law with exponent a , allow degree distribution to change over time, and ask *How should the power law degree exponent $\gamma(N)$ change over time (as the number of nodes N grows) to allow for densification?* We show the following result:

Theorem 3.3.2. *Given a time evolving graph on N nodes that evolves according to Densification Power Law with exponent $a > 1$ and has a Power law degree distribution with exponent $\gamma(N) > 2$, then the degree exponent $\gamma(N)$ evolves with the number of nodes N as*

$$\gamma(N) = \frac{4N^{a-1} - 1}{2N^{a-1} - 1} \quad (3.5)$$

Proof. An undirected graph G on N nodes has $E = \frac{1}{2}N\bar{d}$ edges, where \bar{d} is the average degree in graph G . Then the DPL exponent a is

$$a = \frac{\log(E)}{\log(N)} = \frac{\log(N) + \log(\bar{d}) - \log(2)}{\log(N)} \quad (3.6)$$

In a graph with power law degree distribution, $p(x) = x^{-\gamma}$, with exponent $\gamma > 2$, the average degree \bar{d} is

$$\bar{d} \approx \int_1^\infty xp(x) dx = c \int_1^\infty x^{-\gamma+1} dx = \frac{c}{2-\gamma} x^{-\gamma+2} \Big|_1^\infty = \frac{\gamma-1}{\gamma-2}. \quad (3.7)$$

Now, substituting \bar{d} in equation 3.6 with the result of equation 3.7, and solving for γ , we obtain:

$$\gamma(N) = \frac{4N^{a-1} - 1}{2N^{a-1} - 1} \quad (3.8)$$

□

Here we found the evolution pattern that degree distribution with exponent $\gamma > 2$ has to follow in order to allow for densification. As theorem 3.6 shows the degree distribution has to flatten over time, so that the expected node degree increases, which is the result of densification.

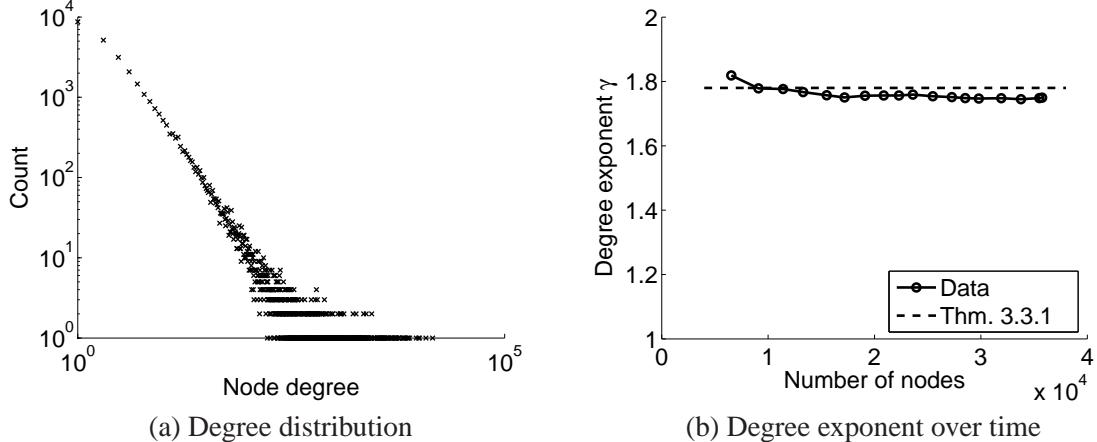


Figure 3.6: Degree distribution (a) and the degree exponent γ over time (b) for the email network EMAIL-INOUT. The network maintains constant slope γ of degree distribution over time. Notice that $\gamma < 2$. We observe a remarkably good agreement between the result of Theorem 3.3.1 (DPL exponent $a = 1.13$), and our measurements (DPL exponent $a = 1.11$) in figure 3.2(e).

Measurements on real networks

Next, given the analysis from the previous section, we went back to the data and checked if graphs we analyzed before behave according to the results of theorems 3.3.1 and 3.3.2.

First, we show an example of a graph where the evolution of the degree distribution and the DPL exponent follow the results of theorem 3.3.1. Using the email network described in section 3.3.1 we found that the degree distribution follows a power law with exponent γ that remains constant over time.

Figure 3.6(a) shows the degree distribution of the email network for last snapshot of the network, *i.e.*, last 2 months of the data. We create the networks by using a 2 month sliding window. We fit the power law degree exponent γ using Maximum Likelihood Estimation (MLE), and plot its evolution over time in figure 3.6(b). Notice γ remains practically constant over time, which is also in agreement with observations reported in [Kossinets and Watts, 2006]. Also notice that the power law degree exponent $\gamma = 1.76 < 2$. Given the degree exponent γ , and using theorem 3.3.1 we obtain the theoretical value of the DPL exponent $a = 2/1.76 \approx 1.13$. The value of DPL exponent we measured in section 3.3 figure 3.2(e) is $a = 1.11$, which is a remarkably good agreement. This shows that there exist graphs in the real world that densify and have decreasing diameter while maintaining constant degree exponent over time.

Last, we show an example of a temporally evolving graph that densifies, and has the power law degree exponent γ changing over time.

Figure 3.7(a) plots the degree distribution of the full HEP-PH citation network from section 3.3.1. In this case the degree distribution only follows a power law in the tail of the distribution, so we applied the following procedure. For every year y , $1992 \leq y \leq 2002$ we create a citation graph and measure the exponent of the power law degree distribution. We apply logarithmic binning and fit the power law degree distribution using MLE on the tail of the degree distribution starting at minimum degree 10. We plot the resulting degree exponent γ over time as a function of the size of the graph in figure 3.7(b).

Using dashed-lines we also plot the degree exponent γ as obtained by theorem 3.3.2. Since the graph does not exhibit power law degree distribution on the entire range, and due to missing past effects, we had to

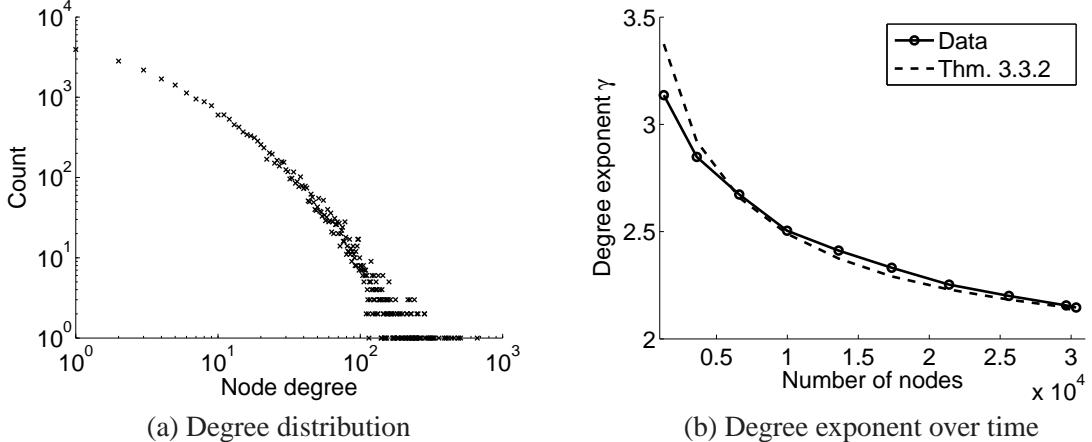


Figure 3.7: Degree distribution (a) and the degree exponent over time (b) for the HEP–PH citation network (CIT-HEP-PH). The network follows power law degree distribution only in the tail. Degree distribution exponent γ is decreasing over time. Notice a good agreement of degree distribution evolution (solid line) as predicted by the theorem 3.3.2 (dashed line).

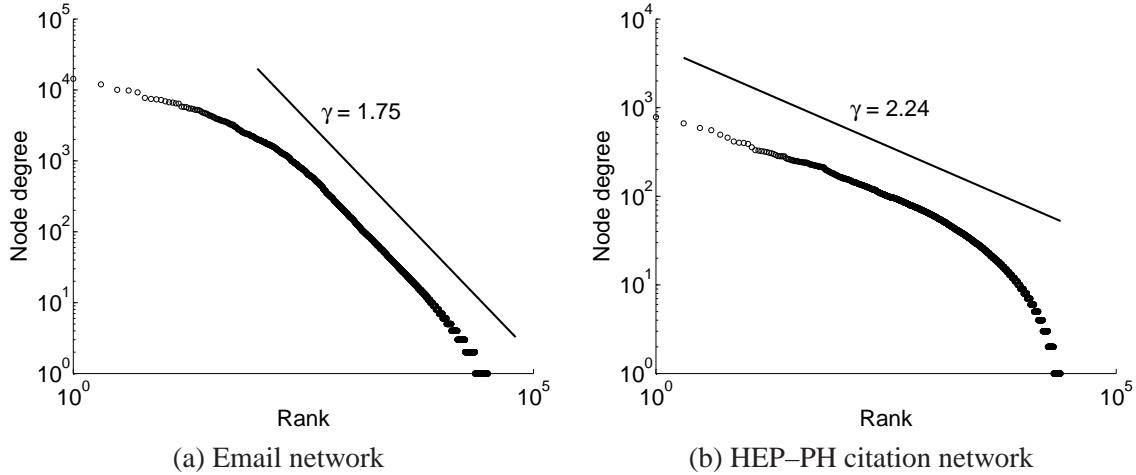


Figure 3.8: Rank Degree plot for the degree distribution of the email, (EMAIL-INOUT) and the HEP–PH (CIT-HEP-PH) networks. We use the same data as in figures 3.6(a) and 3.7(a) but plot node degree vs. rank using the log-log scales. As a eye guideline we plot the solid lines that present the power law decay with exponent $\gamma = 1.75$ and $\gamma = 2.24$, respectively.

appropriately scale time axis with a manually chosen value. Regardless of the manual scaling we think this result indicates that for a class of temporally evolving graphs the degree distribution flattens over time as given by the theorem 3.3.2. This seems to be the case for HEP–PH citation network where the evolution of the degree exponent qualitatively follows the result of theorem 3.3.2.

Figure 3.8 further investigates the degree distribution of the email and HEP–PH networks. We use the same data as in figures 3.6(a) and 3.7(a), and plot the number of nodes of a certain degree against the rank. The solid lines present the power law decay with exponents $\gamma = 1.75$ and $\gamma = 2.24$, respectively. The actual slope of the plotted line is $1/(\gamma - 1)$, which is the relation between the power law exponent γ and the slope of the rank degree plot (see [Adamic, 2000] for more details on these relationships).

In both plots of figure 3.8 we observe linearity which suggests a power law relationship for a part of the degree distribution. For the email network we observe linearity in the tail, and for the HEP–PH citation network in the first part of the distribution. These two plots show that in our two datasets the power law degree distribution does not hold for the entire range. However, we still observe a significant range where power law relationship seems to hold. Regardless of these irregularities there is still very good agreement of the data with the results of theorems 3.3.1 and 3.3.2, which suggests that there exists graphs that densify by maintaining constant power law degree exponent (theorem 3.3.1), and also graphs that densify by degree exponent flattening over time (theorem 3.3.2).

3.4 Proposed models

We have now seen that densification power laws and shrinking effective diameters are properties that hold across a range of diverse networks. Moreover, existing models do not capture these phenomena. We would like to find some simple, local model of behavior, which could naturally lead to the macroscopic phenomena we have observed. We present increasingly sophisticated models, all of which naturally achieve the observed densification; the last one (the “Forest Fire” model) also exhibits shrinking diameter and all the other main patterns known (including heavy-tailed in- and out-degree distributions).

3.4.1 Community Guided Attachment

What are the underlying principles that drive all our observed graphs to obey a densification power law, without central control or coordination? We seek a model in which the densification exponent arises from intrinsic features of the process that generates nodes and edges. While one could clearly define a graph model in which $E(t) \propto N(t)^a$ by simply having each node, when it arrives at time t , generate $N(t)^{a-1}$ out-links — the equivalent of positing that each author of a paper in a citation network has a rule like, “Cite N^{a-1} other documents,” hard-wired in his or her brain — such a model would not provide any insight into the origin of the exponent a , as the exponent is unrelated to the operational details by which the network is being constructed. Instead, our goal is to see how underlying properties of the network evolution process itself can affect the observed densification behavior.

We take the following approach. Power laws often appear in combination with *self-similar* structures. Intuitively, a self-similar object consists of miniature replicas of itself [Schroeder, 1991]. Our approach involves two steps, both of which are based on self-similarity.

We begin by searching for self-similar, recursive structures. In fact, we can easily find several such recursive sets: For example, computer networks form tight groups (*e.g.*, based on geography), which consist of smaller groups, and so on, recursively. Similarly for patents: they also form conceptual groups (“chemistry”, “communications”, etc.), which consist of sub-groups, and so on recursively. Several other graphs feature such “communities within communities” patterns.

For example, it has been argued (see *e.g.* [Watts et al., 2002] and the references therein) that social structures exhibit self-similarity, with individuals organizing their social contacts hierarchically. Moreover, pairs of individuals belonging to the same small community form social ties more easily than pairs of individuals who are only related by membership in a larger community. In a different domain, Menczer studied the frequency of links among Web pages that are organized into a topic hierarchy such as the Open Directory [Menczer, 2002]. He showed that link density among pages decreases with the height of their

least common ancestor in the hierarchy. That is, two pages on closely related topics are more likely to be hyperlinked than are two pages on more distantly related topics.

This is the first, qualitative step in our explanation for the Densification Power Law. The second step is quantitative. We will need a numerical measure of the difficulty in crossing communities. The extent to which it is indeed difficult to form links across communities will be a property of the domain being studied. We call this the *Difficulty Constant*, and we define it more precisely below.

The basic version of the model

We represent the recursive structure of communities-within-communities as a tree Γ , of height H_Γ . We shall show that even a simple, perfectly balanced tree of constant fanout b is enough to lead to a densification power law, and so we will focus the analysis on this basic model.

The nodes \mathcal{V} in the graph we construct will be the leaves of the tree; that is, $N = |\mathcal{V}|$. (Note that $N = b^{H_\Gamma}$.) Let $h_\Gamma(v, w)$ define the standard tree distance of two leaf nodes v and w : that is, $h_\Gamma(v, w)$ is the height of their least common ancestor (the height of the smallest sub-tree containing both v and w).

We will construct a random graph on a set of nodes \mathcal{V} by specifying the probability that v and w form an edge as a function f of $h_\Gamma(v, w)$. We refer to this function f as the *Difficulty Function*. What should be the form of f ? Clearly, it should decrease with h ; but there are many forms such a decrease could take.

The form of f that works best for our purposes comes from the self-similarity arguments we made earlier: We would like f to be scale-free; that is, $f(h)/f(h - 1)$ should be level-independent and thus constant. The only way to achieve level-independence is to define $f(h) = f(0)c^{-h}$. Setting $f(0)$ to 1 for simplicity, we have:

$$f(h) = c^{-h} \quad (3.9)$$

where $c \geq 1$. We refer to the constant c as the *Difficulty Constant*. Intuitively, cross-communities links become harder to form as c increases.

This completes our development of the model, which we refer to as *Community Guided Attachment*: If the nodes of a graph belong to communities-within-communities, and if the cost for cross-community edges is scale-free (Eq. (3.9)), the Densification Power Law follows naturally. No central control or exogenous regulations are needed to force the resulting graph to obey this property. In short, self-similarity itself leads to the Densification Power Law.

Theorem 3.4.1. *In the Community Guided Attachment random graph model just defined, the expected average out-degree \bar{d} of a node is proportional to:*

$$\begin{aligned} \bar{d} &= N^{1-\log_b(c)} && \text{if } 1 \leq c \leq b \\ &= \log_b(N) && \text{if } c = b \\ &= \text{constant} && \text{if } c > b \end{aligned}$$

Proof. For a given node v , the expected out-degree (number of links) \bar{d} of the node is proportional to

$$\bar{d} = \sum_{x \neq v} f(h_\Gamma(x, v)) = \sum_{j=1}^{\log_b(N)} (b-1)b^{j-1}c^{-j} = \frac{b-1}{c} \sum_{j=1}^{\log_b(N)} \left(\frac{b}{c}\right)^{j-1}. \quad (3.10)$$

There are three different cases: if $1 \leq c < b$ then by summing the geometric series we obtain

$$\begin{aligned} \bar{d} &= \frac{b-1}{c} \cdot \frac{\left(\frac{b}{c}\right)^{\log_b(N)} - 1}{\left(\frac{b}{c}\right) - 1} = \left(\frac{b-1}{b-c}\right) (N^{1-\log_b(c)} - 1) \\ &= \Theta(N^{1-\log_b(c)}). \end{aligned}$$

In the case when $c = b$ the series sums to

$$\begin{aligned} \bar{d} &= \sum_{x \neq v} f(h_\Gamma(x, v)) = \frac{b-1}{b} \sum_{j=1}^{\log_b(N)} \left(\frac{b}{b}\right)^{j-1} = \frac{b-1}{b} \log_b(N) \\ &= \Theta(\log_b(N)). \end{aligned}$$

The last case is when Difficulty Constant c is greater than branching factor b ($c > b$), then the sum in Eq. (3.10) converges to a constant even if carried out to infinity, and so we obtain $\bar{d} = \Theta(1)$. \square

Note that when $c < b$, we get a densification law with exponent greater than 1: the expected out-degree is $N(t)^{1-\log_b(c)}$, and so the total number of edges grows as $N(t)^a$ where $a = 2 - \log_b(c)$. Moreover, as c varies over the interval $[1, b)$, the exponent a ranges over all values in the interval $(1, 2]$.

Corollary 3.4.2. *If the Difficulty Function is scale-free ($f(h) = c^{-h}$, with $1 < c < b$), then the Community Guided Attachment obeys the Densification Power Law with exponent*

$$a = 2 - \log_b(c)$$

The Community Guided Attachment model above also leads to some intuitive extreme cases:

- If the cross-community difficulty constant Difficulty Function is too low ($= 1$), then every node can easily connect to every other node, and the average degree \bar{d} is N . That is, we have a near-clique.
- If cross-community difficulty constant is too high then we obtain no densification ($a = 1$), which means that nodes only link inside their own subtree and do not create long range edges to nodes residing in other parts of the tree.

Dynamic Community Guided Attachment

So far we have discussed a model in which nodes are first organized into a nested set of communities, and then they start forming links. We now extend this to a setting in which nodes are added over time, and the nested structure deepens to accommodate them. We will assume that a node only creates out-links at

the moment it is added (and hence, only to nodes already present); this is natural for domains like citation networks in which a paper's citations are written at the same time as the paper itself.

Specifically, the model is as follows. Rather than having graph nodes reside only at the leaves of the tree Γ , there will now be a graph node corresponding to every internal node of Γ as well. Initially, there is a single node v in the graph, and our tree Γ consists just of v . In time step t , we go from a complete b -ary tree of depth $t - 1$ to one of depth t , by adding b new leaves as children of each current leaf. Each of these new leaves will contain a new node of the graph.

Now, each new node forms out-links according to a variant of the process in which all graph nodes are leaves. However, since a new node has the ability to link to internal nodes of the existing tree, not just to other leaves, we need to extend the model to incorporate this. Thus, we define the *tree-distance* $h(v, w)$ between nodes v and w to be the length of a path between them in Γ — this is the length of the path from v up to the least common ancestor of v and w , plus the length of the path from this least common ancestor down to w . Note that if v and w are both leaves, then $h(v, w) = 2h_\Gamma(v, w)$, following our definition of $h_\Gamma(v, w)$ from before.

The process of forming out-links is now as follows: For a constant c , node v forms a link to each node w , independently, with probability $c^{-h(v, w)/2}$. (Note that dividing by 2 in the exponent means this model gives the same probability as basic model in the case when both v and w are leaves.)

Like the first model, this process produces a densification law with exponent $a = 2 - \log_b(c)$ when $c < b$. However, for $c < b^2$, it also yields a heavy-tailed distribution of in-degrees — something that the basic model did not produce. We describe this in the following theorem.

Theorem 3.4.3. *The Dynamic Community Guided Attachment model just defined has the following properties.*

- When $c < b$, the average node degree is $N^{1-\log_b(c)}$ and the in-degrees follow a Zipf distribution with exponent $\frac{1}{2} \log_b(c)$.
- When $b < c < b^2$, the average node degree is constant, and the in-degrees follow a Zipf distribution with exponent $1 - \frac{1}{2} \log_b(c)$.
- When $c > b^2$, the average node degree is constant and the probability of an in-degree exceeding any constant bound k decreases exponentially in k .

Proof. In the proof, all logarithms will be expressed in base b unless specified otherwise.

We begin with the following basic facts. If a node is at height h in the tree, then the number of nodes at distance $d \leq h$ from it is $\Theta(b^d)$. Nodes at distance $d > h$ can be reached by going up for j steps, and then down for $d - j$ steps (if $d - j \leq h + j$). This is maximized for $j = (d - h)/2$, and so the total number of nodes reachable at distance d is $\Theta(b^{(d+h)/2})$.

Case 1: $c < b$ In this case, the expected out-degree for a leaf node is

$$\sum_{d=0}^{2 \log N} \Theta\left(\frac{b^{d/2}}{c^{d/2}}\right) = \Theta\left(\frac{b^{\log N}}{c^{\log N}}\right) = \Theta\left(\frac{N}{c^{\log N}}\right) = \Theta\left(N^{1-\log c}\right).$$

Since the expected out-degree values for other nodes are smaller, and since a constant fraction of all nodes are leaves, it follows that the expected value of the out-degree taken over all nodes is $\Theta(N^{1-\log c})$ as well.

Now we compute the expected in-degree of a node at height h . This is

$$\sum_{d \leq h} \Theta\left(\frac{b^d}{c^{d/2}}\right) + \sum_{d > h} \Theta\left(\frac{b^{(d+h)/2}}{c^{d/2}}\right) = \sum_{d \leq h} \Theta\left(\frac{b^{d/2}}{c^{d/2}}\right) b^{d/2} + \sum_{d > h} \Theta\left(\frac{b^{d/2}}{c^{d/2}}\right) b^{h/2}.$$

The largest term in this sum is the last, for $d = 2 \log N - h$. Here it takes the value

$$\Theta\left(\frac{b^{\log N}}{c^{\log N - (h/2)}}\right) = \Theta\left(\frac{b^{\log N}}{c^{\log N}}\right) c^{h/2} = \Theta\left(N^{1-\log c} c^{h/2}\right).$$

The maximum expected in-degree z is achieved for $h = \log N$, when we get

$$z = \Theta\left(N^{1-\log c} c^{.5 \log N}\right) = \Theta\left(N^{1-.5 \log c}\right).$$

So for a node at depth $t = \log N - h$, we get an expected in-degree of

$$\Theta\left(N^{1-\log c} c^{(\log N - t)/2}\right) = \Theta\left(z c^{-t/2}\right).$$

Hence, to compute a Zipf exponent, we see that a node of degree rank $r = b^t$ has depth t , so it has degree

$$\Theta\left(\frac{z}{c^{t/2}}\right) = \Theta\left(\frac{z}{r^{.5 \log c}}\right).$$

Case 2: $b < c < b^2$ In this case, the expected out-degree for a leaf node is

$$\sum_{d=0}^{2 \log N} \Theta\left(\frac{b^{d/2}}{c^{d/2}}\right) = \Theta(1).$$

Since the expected out-degree values for other nodes are smaller, it follows that the expected value of the out-degree taken over all nodes is $\Theta(1)$ as well.

Now we compute the expected in-degree of a node at height h . This is

$$\sum_{d \leq h} \Theta\left(\frac{b^d}{c^{d/2}}\right) + \sum_{d > h} \Theta\left(\frac{b^{(d+h)/2}}{c^{d/2}}\right) = \sum_{d \leq h} \Theta\left(\frac{b^{d/2}}{c^{d/2}}\right) b^{d/2} + \sum_{d > h} \Theta\left(\frac{b^{d/2}}{c^{d/2}}\right) b^{h/2}.$$

Since $b < c < b^2$, these terms increase geometrically up to $d = h$, then decrease. Thus, the largest term is for $d = h$, where it is $\Theta(b^h c^{-h/2})$.

Thus the maximum degree is $z = \Theta(N^{1-.5 \log c})$, and for depth $t = \log N - h$, we get a degree of

$$\Theta\left(\left(\frac{b}{c^{1/2}}\right)^{\log N} \left(\frac{b}{c^{1/2}}\right)^{-t}\right) = \Theta\left(z \left(\frac{b}{c^{1/2}}\right)^{-t}\right).$$

Now, $b/c^{1/2} = b^{1-.5 \log c}$, so a node of degree rank $r = b^t$ (at depth t) has degree $\Theta(z/r^{1-.5 \log c})$.

Case 3: $c > b^2$ The expected out-degrees here are only smaller than they are in the previous case, and hence the expected value of the out-degree taken over all nodes is $\Theta(1)$.

The node whose in-degree is most likely to exceed a fixed bound k is the root, at height $h = \log N$. The in-degree of the root is a sum X of independent 0-1 random variables X_v , where X_v takes the value 1 if node v links to the root, and X_v takes the value 0 otherwise. We have

$$EX = \sum_v EX_v = \sum_{d \leq \log N} \Theta\left(\frac{b^d}{c^{d/2}}\right) = \Theta(1),$$

and hence by Chernoff bounds, the probability that it exceeds a given value $k > EX$ decreases exponentially in k . \square

Thus, the dynamic Community Guided Attachment model exhibits three qualitatively different behaviors as the parameter c varies: densification with heavy-tailed in-degrees; then constant average degree with heavy-tailed in-degrees; and then constant in- and out-degrees with high probability. Note also the interesting fact that the power law degree exponent is maximized for the value of c right at the onset of densification.

Finally, we have experimented with versions of the dynamic Community Guided Attachment model in which the tree is not balanced, but rather deepens more on the left branches than the right (in a recursive fashion). We have also considered versions in which a single graph node can “reside” at two different nodes of the tree Γ , allowing for graph nodes to be members of different communities. Experimental results and overall conclusions were all the time the same and consistent regardless of the particular version (modification) of the dynamic Community Guided Attachment model used.

3.4.2 The Forest Fire Model

Community Guided Attachment and its extensions show how densification can arise naturally, and even in conjunction with heavy-tailed in-degree distributions. However, it is not a rich enough class of models to capture all the properties in our network datasets. In particular, we would like to capture both the shrinking effective diameters that we have observed, as well as the fact that real networks tend to have heavy-tailed out-degree distributions (though generally not as skewed as their in-degree distributions). The Community Guided Attachment models do not exhibit either of these properties.

Specifically, our goal is as follows. Given a (possibly empty) initial graph G , and a sequence of new nodes $v_1 \dots v_N$, we want to design a simple randomized process to successively link v_i to nodes of G ($i = 1, \dots, N$) so that the resulting graph G_{final} will obey all of the following patterns: heavy-tailed distributions for in- and out-degrees, the Densification Power Law, and shrinking effective diameter.

We are guided by the intuition that such a graph generator may arise from a combination of the following components:

- some type of “rich get richer” attachment process, to lead to heavy-tailed in-degrees;
- some flavor of the “copying” model [Kumar et al., 2000], to lead to communities;
- some flavor of Community Guided Attachment, to produce a version of the Densification Power Law;
- and a yet-unknown ingredient, to lead to shrinking diameters.

Note that we will *not be assuming a community hierarchy on nodes*, and so it is not enough to simply vary the Community Guided Attachment model.

Based on this, we introduce the *Forest Fire Model*, which is capable of producing all these properties. To set up this model, we begin with some intuition that also underpinned Community Guided Attachment: nodes arrive in over time; each node has a “center of gravity” in some part of the network; and its probability of linking to other nodes decreases rapidly with their distance from this center of gravity. However, we add to this picture the notion that, occasionally, a new node will produce a very large number of out-links. Such nodes will help cause a more skewed out-degree distribution; they will also serve as “bridges” that connect formerly disparate parts of the network, bringing the diameter down.

The Basic Forest Fire Model

Following this plan, we now define the most basic version of the model. Essentially, nodes arrive one at a time and form out-links to some subset of the earlier nodes; to form out-links, a new node v attaches to a node w in the existing graph, and then begins “burning” links outward from w , linking with a certain probability to any new node it discovers. One can view such a process as intuitively corresponding to a model by which an author of a paper identifies references to include in the bibliography. He or she finds a first paper to cite, chases a subset of the references in this paper (modeled here as random), and continues recursively with the papers discovered in this way. Depending on the bibliographic aids being used in this process, it may also be possible to chase back-links to papers that cite the paper under consideration. Similar scenarios can be considered for social networks: a new computer science (CS) graduate student arrives at a university, meets some older CS students, who introduce him/her to their friends (CS or non-CS), and the introductions may continue recursively.

We formalize this process as follows, obtaining the Forest Fire Model. To begin with, we will need two parameters, a *forward burning probability* p , and a *backward burning ratio* r , whose roles will be described below. Consider a node v joining the network at time $t > 1$, and let G_t be the graph constructed thus far. (G_1 will consist of just a single node.) Node v forms out-links to nodes in G_t according to the following process.

- (i) v first chooses an *ambassador node* w uniformly at random, and forms a link to w .
- (ii) We generate two random numbers: x and y that are geometrically distributed with means $p/(1 - p)$ and $rp/(1 - rp)$ respectively. Node v selects x out-links and y in-links of w incident to nodes that were not yet visited. Let w_1, w_2, \dots, w_{x+y} denote the other ends of these selected links. If not enough in- or out-links are available, v selects as many as it can.

- (iii) v forms out-links to w_1, w_2, \dots, w_{x+y} , and then applies step (ii) recursively to each of the nodes w_1, w_2, \dots, w_{x+y} . As the process continues, nodes cannot be visited a second time, preventing the construction from cycling.

Thus, the “burning” of links in Forest Fire model begins at w , spreads to w_1, \dots, w_{x+y} , and proceeds recursively until it dies out. In terms of the intuition from citations in papers, the author of a new paper v initially consults w , follows a subset of its references (potentially both forward and backward) to the papers w_1, \dots, w_{x+y} , and then continues accumulating references recursively by consulting these papers. The key property of this model is that certain nodes produce large “conflagrations,” burning many edges and hence forming many out-links before the process ends.

Despite the fact that there is no explicit hierarchy in the Forest Fire Model, as there was in Community Guided Attachment, there are some subtle similarities between the models. Where a node in Community Guided Attachment was the child of a parent in the hierarchy, a node v in the Forest Fire Model also has an “entry point” via its chosen ambassador node w . Moreover, just as the probability of linking to a node in Community Guided Attachment decreased exponentially in the tree distance, the probability that a new node v burns k successive links so as to reach a node u lying k steps away is exponentially small in k . (Of course, in the Forest Fire Model, there may be many paths that could be burned from v to u , adding some complexity to this analogy.)

In fact, our Forest Fire Model combines the flavors of several older models, and produces graphs qualitatively matching their properties. We establish this by simulation, as we describe below, but it is also useful to provide some intuition for why these properties arise.

- *Heavy-tailed in-degrees.* Our model has a “rich get richer” flavor: highly linked nodes can easily be reached by a newcomer, no matter which ambassador it starts from.
- *Communities.* The model also has a “copying” flavor: a newcomer copies several of the neighbors of his/her ambassador (and then continues this recursively).
- *Heavy-tailed out-degrees.* The recursive nature of link formation provides a reasonable chance for a new node to burn many edges, and thus produce a large out-degree.
- *Densification Power Law.* A newcomer will have a lot of links near the community of his/her ambassador; a few links beyond this, and significantly fewer farther away. Intuitively, this is analogous to the Community Guided Attachment, although without an explicit set of communities.
- *Shrinking diameter.* It is not a priori clear why the Forest Fire Model should exhibit a shrinking diameter as it grows. Graph densification is helpful in reducing the diameter, but it is important to note that densification is certainly not enough on its own to imply shrinking diameter. For example, the Community Guided Attachment model obeys the Densification Power Law, but our experiments also show that the diameter slowly increases (not shown here).

Rigorous analysis of the Forest Fire Model appears to be quite difficult. However, in simulations, we find that by varying just the two parameters p and r , we can produce graphs that densify ($a > 1$), exhibit heavy-tailed distributions for both in- and out-degrees (Fig. 3.10), and have diameters that decrease. This is illustrated in Figure 3.9, which shows plots for the effective diameter and the Densification Power Law exponent as a function of the number of nodes for some selections of p and r .

We see that depending on the forward and backward burning parameters the Forest Fire Model is capable of generating sparse or dense graphs with effective diameters that either increase or decrease, while also

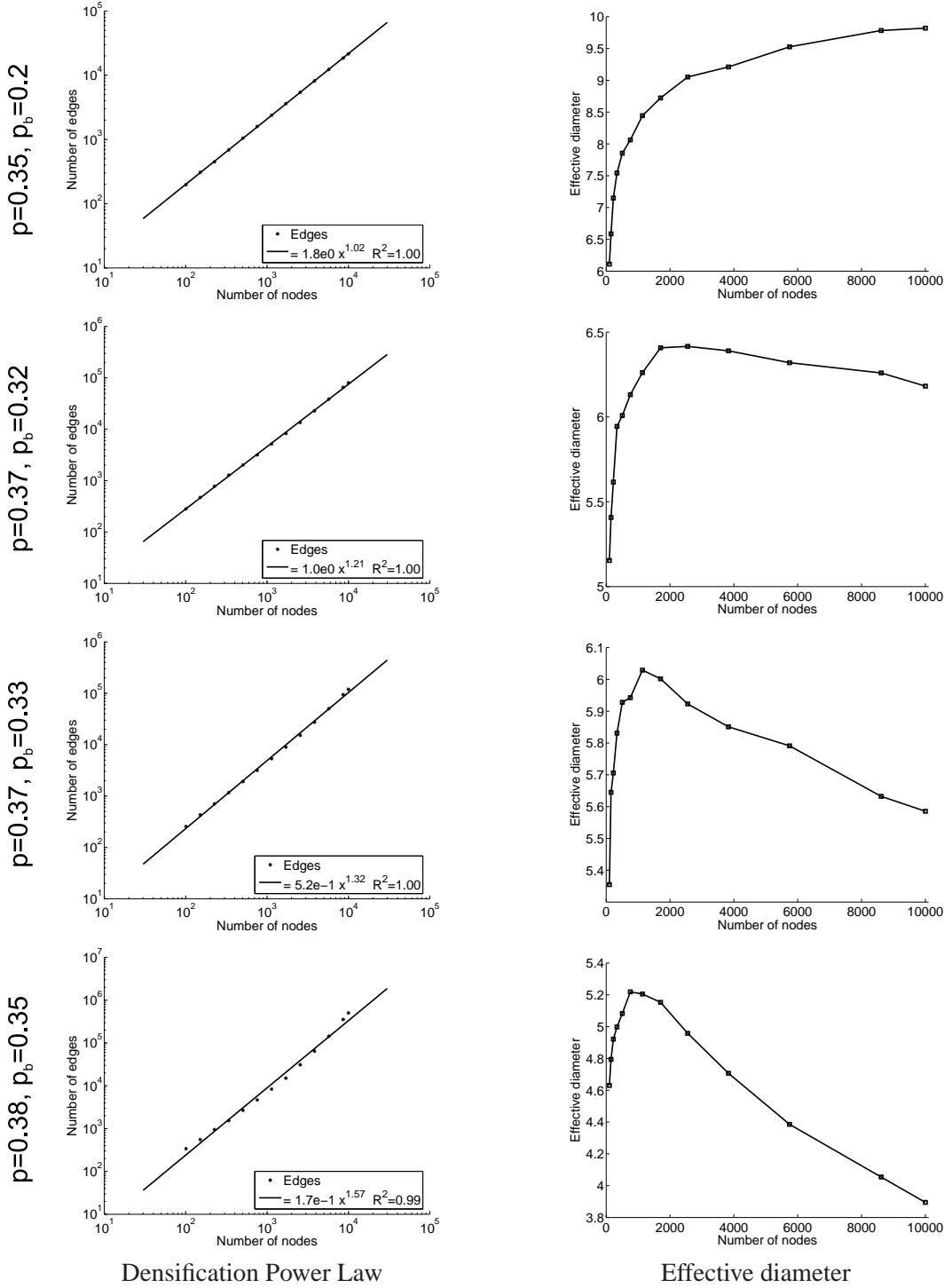


Figure 3.9: The DPL plot and the effective diameter for the Forest Fire model. Row 1: sparse graph ($a = 1.01 < 2$), with increasing diameter (forward burning probability $p = 0.35$, backward probability $p_b = 0.20$). Row 2: (most realistic case:) densifying graph ($a = 1.21 < 2$) with slowly decreasing diameter ($p = 0.37, p_b = 0.32$). Row 3: densifying graph ($a = 1.32 < 2$) with decreasing diameter ($p = 0.37, p_b = 0.33$). Row 4: dense graph with densification exponent close to 2 ($a = 1.57$) and decreasing diameter ($p = 0.38, p_b = 0.35$).

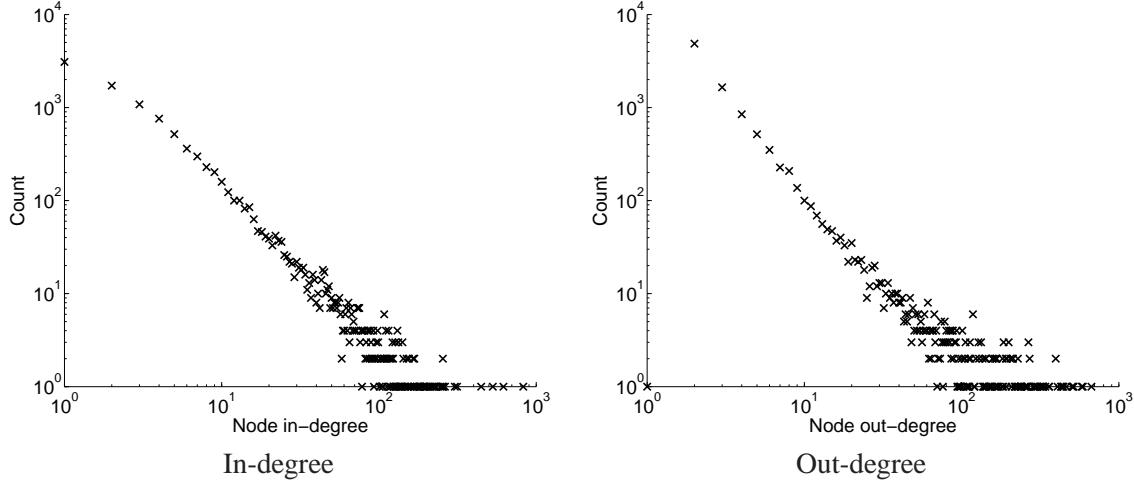


Figure 3.10: Degree distribution of a sparse graph with decreasing diameter (forward burning probability: 0.37, backward probability: 0.32).

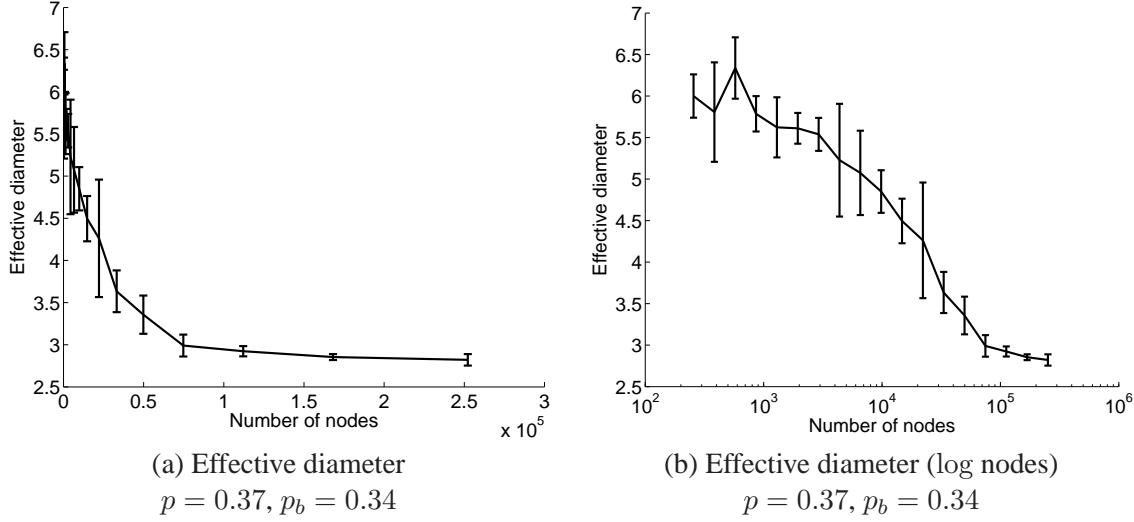


Figure 3.11: Evolution of effective diameter of Forest Fire model while generating a large graph. Both plots show the same data; left one plots on linear scales and the right one plots on log-linear scales (effective diameter vs. log number of nodes). Error bars show the confidence interval of the estimated effective diameter. Notice that the effective diameter shrinks and then slowly converges.

producing power law in- and out-degree distributions (figure 3.10). Informally, a dense graph has close to a linear number of edges incident to each node, while a sparse graph has significantly fewer than a linear number of edges incident to each node.

Also notice the high sensitivity of the parameter space. We fix the forward burning probability p , and by increasing the backward burning probability p_b ($p_b = r \cdot p$) for only a few percent we move from an increasing to a slowly and then to more rapidly decreasing effective diameter (figure 3.9).

Figure 3.11 plots the evolution of the effective diameter of Forest Fire. We generated a single large graph

on 250,000 nodes and measured the effective diameter over time. Error bars present 1 standard deviation of the estimated effective diameter over 10 runs. Both plots show the same data. The left figure plots the number of nodes on linear while the right plots the log number of nodes. Notice the convergence of the effective diameter. At first it shrinks more rapidly and then slowly converges to a low value.

Extensions to the Forest Fire Model

Our basic version of the Forest Fire Model exhibits rich structure with just two parameters. By extending the model in natural ways, we can fit observed network data even more closely. We propose two natural extensions: “*orphans*” and multiple ambassadors.

“*Orphans*”: In both the patent and arXiv citation graphs, there are many isolated nodes, that is, documents with no citations into the corpus. For example, many papers in the arXiv only cite non-arXiv papers. We refer to them as *orphans*. Our basic model does not produce orphans, since each node always links at least to its chosen ambassador. However, it is easy to incorporate orphans into the model in two different ways. We can start our graphs with $n_0 > 1$ nodes at time $t = 1$; or we can have some probability $q > 0$ that a newcomer will form no links (not even to its ambassador) and so become an orphan.

We find that such variants of the model have a more pronounced decrease in the effective diameter over time, with large distances caused by groups of nodes linking to different orphans gradually diminishing as further nodes arrive to connect them together.

Multiple ambassadors: We experimented with allowing newcomers to choose more than one ambassador with some positive probability. That is, rather than burning links starting from just one node, there is some probability that a newly arriving node burns links starting from two or more. This extension also accentuates the decrease in effective diameter over time, as nodes linking to multiple ambassadors serve to bring together formerly far-apart parts of the graph.

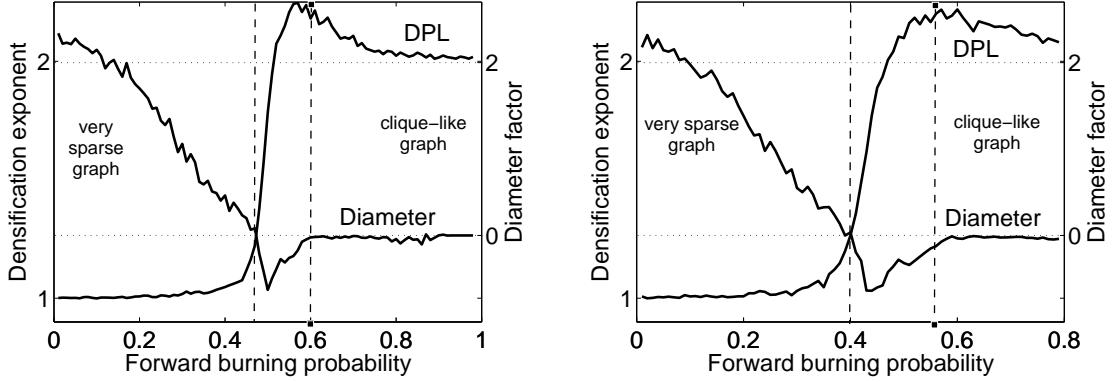
Burning a fixed percentage of neighbors: We also considered a version of Forest Fire where the fire burns a fixed percentage of node’s edges, *i.e.*, the number of burned edges is proportional to the node’s degree. When a fire comes into a node, for each unburned neighbor we *independently* flip a biased coin to determine where to spread the fire. This continues recursively until no new nodes are burned. In case of forward and backward burning probabilities we have two coins, one for out- and one for in-edges.

The problem with this version of the model is that, once there is a single large fire that burns a large fraction of the graph, many subsequent fires will also burn much of the graph. This results in a bell-shaped, non-heavy-tailed degree distribution and gives two regimes of densification — slower densification before the first big fire, and quadratic ($\alpha = 2$) densification afterwards.

We also experimented with the model where burning probability decayed exponentially as the fire moves away from the ambassador node.

Phase plot

In order to understand the densification and the diameter properties of graphs produced by the Forest Fire Model, we explored the full parameter space of the basic model in terms of the two underlying parameters: the forward burning probability p and the backward burning ratio r .



(a) We fix burning ratio, $r = 0.5$
and vary forward burning probability p

(b) We fix backward burning prob., $p_b = 0.3$
and vary forward burning probability p

Figure 3.12: We vary the forward burning probability while fixing burning ratio (a) or backward burning probability (b). The plot gives a very precise cut through Forest Fire parameter space. Notice that each plot has *two* vertical axes: DPL exponent on the left, and the diameter log-fit factor on the right. Observe a very sharp transition in DPL exponent and a narrow region, indicated by vertical dashed lines, where Forest Fire produces slowly densifying graphs with decreasing effective diameter.

Note, there are two equivalent ways to parameterize the Forest Fire model. We can use the forward burning probability p and the backward burning ratio r ; or the forward burning probability p and the backward burning probability p_b ($p_b = rp$). We examine both and show two cuts through the parameter space.

Figure 3.12 shows how the densification exponent and the effective diameter depend on forward burning probability p . In the left plot of figure 3.12 we fix the backward burning ratio $r = 0.5$, and in the right plot we fix the backward burning probability $p_b = 0.3$. We vary forward burning probability, and plot the Densification Power Law exponent. The densification exponent a is computed as in Section 3.3, by fitting a relation of the form $E(t) \propto N(t)^a$. Notice the very sharp transition between the regimes with no densification and those with very high densification.

On the same plot we also show the *Effective diameter log-fit factor* α . We fit a logarithmic function of the form $D^*(t) = \alpha \log t + \beta$ (where t is the current time, and hence the current number of vertices) to the last half of the effective diameter plot; we then report the factor α . Thus, Diameter Factor $\alpha < 0$ corresponds to decreasing effective diameter over time, and $\alpha > 0$ corresponds to increasing effective diameter.

Going back to Figure 3.12, notice that at low values of forward burning probability p , we observe increasing effective diameter and no densification ($a = 1$). As p increases, the effective diameter grows slower and slower. For a narrow band of p we observe *decreasing effective diameter*, negative α (the small valley around $p = 0.45$). With high values of p the effective diameter is constant ($\alpha \approx 0$), which means that the generated graph is effectively a clique with effective diameter close to 1 and DPL exponent $a \approx 2$. Also notice that the sharp transition in the DPL exponent and the decreasing effective diameter are very well aligned.

This simulations indicate that even the basic Forest Fire Model is able to produce sparse and slowly densifying (with densification exponent near 1) graphs in which the effective diameter decreases.

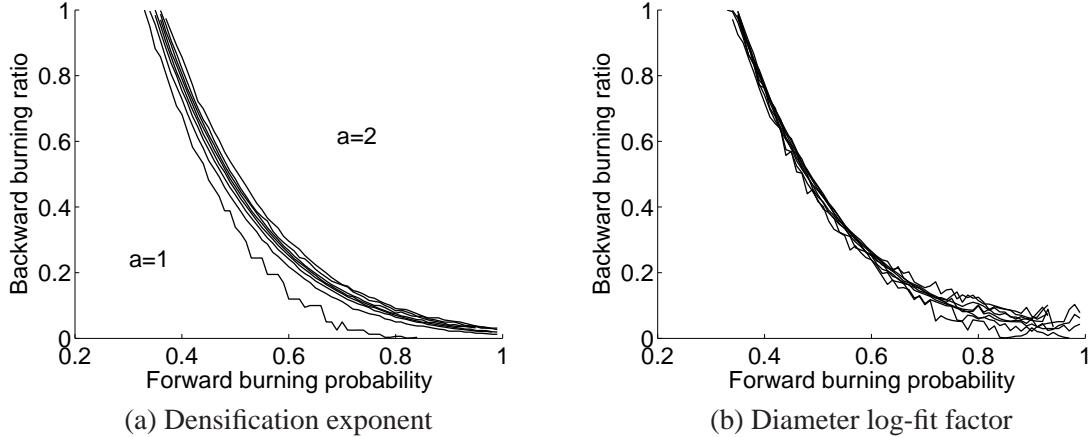


Figure 3.13: Contour plots: The Densification Power Law exponent a (left) and the effective diameter log-fit factor α (right) over the parameter space (forward-burning probability and backward burning ratio) of the Forest Fire model.

Figure 3.13 shows how the densification exponent and the effective diameter depend on the values of the Forest Fire parameters p and r .

Figure 3.13(a) gives the contour plot of the densification exponent a . The lower left part corresponds to $a = 1$ (the graph maintains constant average degree), and in the upper right part $a = 2$ – the graph is “dense”, that is, the number of edges grows quadratically with the number of nodes, as, e.g., in the case of a clique. The contours in-between correspond to 0.1 increase in DPL exponent: the left-most contour corresponds to $a = 1.1$ and the right-most contour corresponds to $a = 1.9$. The desirable region is in-between; we observe that it is very narrow: a increases dramatically along a contour line, suggesting a sharp transition.

Figure 3.13(b) gives the contour plot for the Effective diameter log-fit factor α as defined above. Each contour correspond to diameter factor α . We vary α in range $-0.3 \leq \alpha \leq 0.1$, with step-size 0.05. Notice, the boundary in parameter space between decreasing and increasing effective diameter is very narrow.

Do contour plots of Densification Power Law and Shrinking Diameters from Figure 3.13 follow the same shape? More exactly, does the boundary between decreasing and increasing diameters follow the same shape as the transition in the densification exponent?

We answer this question on figure 3.14, where we superimpose phase contours of DPL and the effective diameter over the Forest Fire parameter space. The left plot superimposes phase contours for the Densification Power Law exponent $a = 1.3$ and the diameter log-fit factor $\alpha = -0.05$. The right plot superimposes contours for $a = 1.6$ and $\alpha = -0.30$. In both cases we observe very good alignment of the two phase lines which suggests the same shape of the transition boundary for the Densification Power Law exponent and the Effective Diameter.

We also observe similar behavior with orphans and multiple ambassadors. These additional features in the model help further separate the diameter decrease/increase boundary from the densification transition, and so widen the region of parameter space for which the model produces reasonably sparse graphs with decreasing effective diameters.

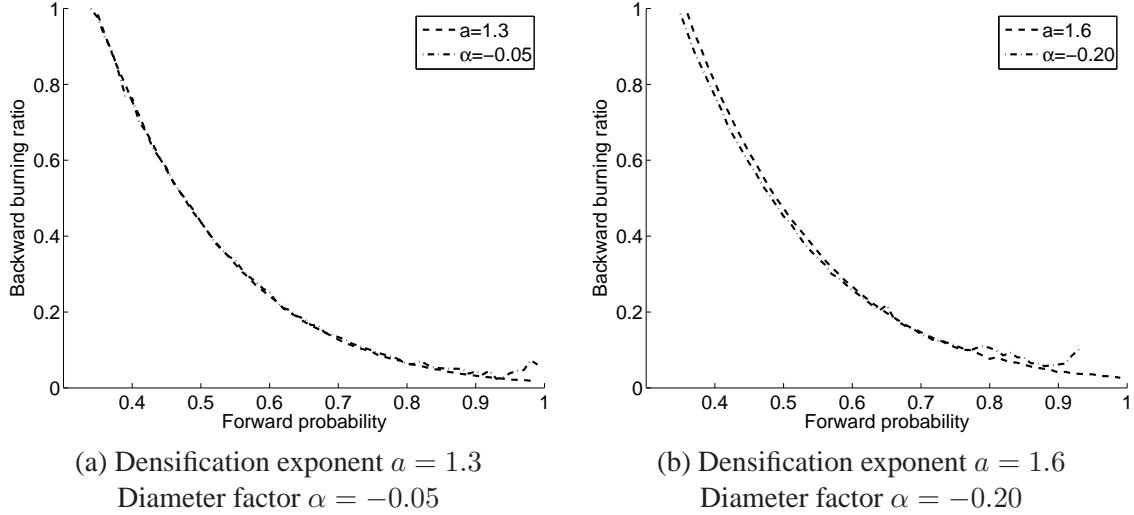


Figure 3.14: We superimpose the Densification Power Law exponent a and diameter log-fit α factor over the Forest Fire Model parameter space. Notice that the shape of transition boundary of the densification and the shrinking diameter very much follow the same shape.

3.5 Discussion

Despite the enormous recent interest in large-scale network data, and the range of interesting patterns identified for static snapshots of graphs (*e.g.*, heavy-tailed distributions, small-world phenomena), there has been relatively little work on the properties of the time evolution of real graphs. This is exactly the focus of this work. The main findings and contributions follow:

- The Densification Power Law: In contrast to the standard modeling assumption that the average out-degree remains constant over time, we discover that real graphs have out-degrees that grow over time, following a natural pattern (Eq. (3.1)).
- Shrinking diameters: Our experiments also show that the standard assumption of slowly growing diameters does not hold in a range of real networks; rather, the diameter may actually exhibit a gradual decrease as the network grows.
- We show that our Community Guided Attachment model leads to the Densification Power Law, and that it needs only one parameter to achieve it.
- We give the Forest Fire model, based on only two parameters, which is able to capture patterns observed both in previous work and in the current study: heavy-tailed in- and out-degrees, the Densification Power Law, and a shrinking diameter.
- We notice that the Forest Fire Model exhibits a sharp transition between sparse graphs and graphs that are densifying. Graphs with decreasing effective diameter are generated around this transition point.
- Finally, we find a fundamental relation between the temporal evolution of the graph's power law degree distribution and the Densification Power Law exponent. We also observe that real datasets exhibit this type of relation.

Our work here began with an investigation of the time-evolution of a set of large real-world graphs across diverse domains. It resulted in the finding that real-world graphs are becoming denser as they grow, and that in many cases their effective diameters are decreasing. This challenges some of the dominant assumptions in recent work on random graph models, which assumes constant (or at most logarithmic) node degrees, and diameters that increase slowly in the number of nodes. Building on these findings, we have proposed a set of simple graph generation processes, capable of producing graphs that exhibit densification and exhibit decreasing effective diameter.

Our results have potential relevance in multiple settings, including 'what if' scenarios; in forecasting of future parameters of computer and social networks; in anomaly detection on monitored graphs; in designing graph sampling algorithms; and in realistic graph generators.

We just examined the evolution of macroscopic statistical properties of networks by studying a set of snapshots. Next, we continue examining network evolution but at much finer granularity. We examine evolution of the online social networks by studying individual edge arrivals from the first to the "million-th" edge.

Chapter 4

Microscopic network evolution

In this chapter we present a microscopic analysis of the edge-by-edge evolution of four large online social networks. The use of the maximum-likelihood principle allows us to quantify the bias of new edges towards the degree and age of nodes, and to objectively compare various models such as preferential attachment. In fact, our work is the first to directly quantify the amount of preferential attachment in large social networks.

Our study shows that most new edges span very short distances, typically closing triangles. Motivated by these observations, we develop a complete model of network evolution, incorporating node arrivals, edge initiation, and edge destination selection processes. While node arrivals are mostly network-specific, the edge initiation process can be captured by exponential node lifetimes and a “gap” model based on a power law with exponential cutoff. We arrive at an extremely simple yet surprisingly accurate description of the edge destination selection in real networks. Our model of network evolution can be used to generate arbitrary-sized synthetic networks that closely mimic the macroscopic characteristics of real social networks.

4.1 Introduction

In recent years a wide variety of models have been proposed for the growth of complex networks. These models are typically advanced in order to reproduce statistical network properties observed in real-world data. They are evaluated on the fidelity with which they reproduce these global network statistics and patterns. In many cases, the goal is to define individual node behaviors that result in a global structure such as power law node degree distributions; in other cases, the goal is to match some other network property such as small diameter.

For example, the observation of heavy-tailed degree distributions [Faloutsos et al., 1999] led to hypothesis about edge creation processes (*e.g.*, preferential attachment [Barabási and Albert, 1999]) that could lead to this observation. In fact, there are several edge creation processes that all lead to heavy-tailed degree distributions and it is not clear which among them captures reality best.

Here we take a different approach. Instead of only focusing on the global network structure and then hypothesizing about what kind of microscopic node behavior would reproduce the observed macroscopic network structure, we focus *directly* on the microscopic node behavior *per se*. For the first time at such

a large scale, we study a sequence of millions of individual edge arrivals, which allows us to directly evaluate and compare microscopic processes that give rise to global network structure.

4.1.1 Evaluation based on likelihood

Given that the microscopic behavior of nodes solely determines the macroscopic network properties, a good network model should match real-world data on global statistics, while maximizing the likelihood of the low-level processes generating the data. Towards this goal, we propose the use of model likelihood of individual edges as a way to evaluate and compare various network evolution models.

Likelihood has not been considered to date in the analysis of the evolution of large social networks mainly due to lack of data and computational issues. Many early network datasets contained only a single or a small number of snapshots of the data, making likelihood computations for evolutionary models infeasible. In contrast, we study four large social networks with *exact* temporal information about individual arrivals of millions of nodes and edges. Here we are therefore able to consider edge-by-edge evolution of networks from their inception onwards, and hence efficiently compute the likelihood that a particular model would have produced a particular edge, given the current state of the network. In contrast to previous work on evolution of large networks that used a series of snapshots to consider patterns at global scale, we study the exact edge arrival sequence, which means we are able to *directly* observe and model the fine-grained network evolutionary processes that are directly responsible for global network patterns and statistics.

A likelihood-based approach has several advantages over approaches based purely on global statistics:

- (1) Models may be compared directly in a unified way, rather than arguing whether faithful reproduction of, *e.g.*, diameter is more important than clustering coefficient and so forth.
- (2) As our understanding of real-world networks improves, the evaluation criterion, *i.e.*, likelihood, remains unchanged while the generative models improve to incorporate the new understanding. Success in modeling can therefore be effectively tracked.
- (3) Models may be meaningfully distinguished based on as-yet-undiscovered properties of real-world data.

4.1.2 Data and model structure

We consider four large online social network datasets — FLICKR (flickr.com, a photo-sharing website), DELICIOUS (del.icio.us, a collaborative bookmark tagging website), YAHOO! ANSWERS (answers.yahoo.com, a knowledge sharing website), and LINKEDIN (linkedin.com, a professional contacts website) — where nodes represent people and edges represent social relationships. In all networks all personally identifiable data was hashed and nodes were assigned random ids.

These networks are large with up to millions of nodes and edges, and the time span of the data ranges from four months to almost four years. All the networks are in early stages of their evolution with the connected component being small and the clustering coefficient increasing over time.

We consider models that can be decomposed into three core processes that completely describe the evolution of the network:

- (1) the *node arrival process* that governs the arrival of new nodes into the network,
- (2) the *edge initiation process* that determines for each node when it will initiate a new edge, and
- (3) the *edge destination selection process* that determines the destination of a newly initiated edge.

Our networks do not include removal of nodes or edges, so we do not model deletion (although we do model the “death” of a node in the sense that it ceases producing new edges).

4.1.3 Our results

We begin with a series of analyses of our four networks, capturing the evolution of key network parameters, and evaluation of the extent to which the edge destination selection process subscribes to preferential attachment. We show that the inherently non-local nature of preferential attachment is fundamentally unable to capture important characteristics in these networks. To the best of our knowledge, this is the first direct large-scale validation of the preferential attachment model in social networks.

Next, we provide a detailed analysis of the data in order to consider parsimonious models for edge destination selection that incorporate locality. We evaluate a wide variety of such models using the maximum-likelihood principle and choose a simple triangle-closing model that is free of parameters. Based on the findings, we then propose a complete network evolution model that accurately captures a variety of network properties. We summarize our model based on the three processes listed earlier.

- (1) *Node arrival process*: We find large variation in node arrival rates over the four networks, ranging from exponential to sub-linear growth. Thus we treat node arrival rate as input to our model.
- (2) *Edge initiation process*: Upon arrival, a node draws its lifetime and then keeps adding edges until reaching its lifetime, with edges inter-arrival rate following a power law with exponential cut-off distribution. We find that edge initiations are *accelerating* with node degree (age), and prove that this leads to power law out degree distributions. The model produces accurate fits and high likelihood.
- (3) *Edge destination selection process*: We find that most edges (30%–60%) are local as they close triangles, *i.e.*, the destination is only two hops from the source. We consider a variety of triangle-closing mechanisms and show that a simple scheme, where a source node chooses an intermediate node uniformly from among its neighbors, and then the intermediate node does the same, has high likelihood.

This scheme is easily and naturally expanded to capture non-local edges according to the distribution of source-destination distance observed in all networks.

Our model is simple and easy to implement. It precisely defines the network evolution process, and we also give parameter settings that allow others to generate networks at arbitrary scale or to take a current existing network and further evolve it. We show that our model produces realistic social network evolution following the true evolution of network properties such as clustering coefficient and diameter; our purely local model gives rise to accurate global properties.

Moreover, our model is also complete. In contrast to Preferential Attachment [Albert et al., 1999], Copying model [Kumar et al., 2000] or Forest Fire model [Leskovec et al., 2005b] where nodes arrive one at a time, immediately create all their edges and then essentially die, our model describes the evolution much more precisely as in our model nodes appear, create one edge at a time, then go to sleep, wake up, create next edge and so on until they die.

4.2 Relation to previous work on network evolution

Many studies on online social networks, world wide web, and biological networks focused on macroscopic properties of static or evolving networks such as degree distributions, diameter, clustering coefficient, communities, densification and shrinking diameters [Faloutsos et al., 1999, Albert and Barabási, 2002, Strogatz, 2001, Newman, 2003, Dorogovtsev and Mendes, 2003, Broder et al., 2000, Fetterly et al., 2004, Leskovec et al., 2007b, Ntolas et al., 2004, Kumar et al., 2006]. In contrast the following chapter focuses on local microscopic processes that give raise to observed macroscopic network properties, like heavy tailed degree distributions or densification.

Recently, researchers examined the finer aspects of edge creation by focusing on a small set of network snapshots. The role of common friends in community formation was analyzed by Backstrom et al. [Backstrom et al., 2006]. A similar study on the collaboration between scientists was done by Newman [Newman, 2001]. Kleinberg and Liben-Nowell [Liben-Nowell and Kleinberg, 2003] studied the predictability of edges in social networks. Later on Capocci *et al.* [Capocci et al., 2006] focused on preferential attachment mechanism in Wikipedia. However, they used a series of weekly snapshots of Wikipedia, while our results are much more precise as we use the exact edge arrival sequence. They observed the (sublinear) preferential attachment up to page degree $d \approx 100$ and for $d > 100$ linking probability actually decreased with d .

The role of triangle closure in small social networks was long studied by sociologists, but never on such a large scale. Simmel theorized that people with common friends are more likely to create friendships and Krackhardt and Handcock [Krackhardt and Handcock, 2007] applied this theory to explain the evolution of triangle closures. A network model based on closed triangles was proposed by Shi et al. [Shi et al., 2007].

The maximum-likelihood principle that will be a common theme throughout the chapter has been typically used to estimate network model parameters [Wasserman and Pattison, 1996, Leskovec and Faloutsos, 2007, Wiuf et al., 2006] or for model selection [Bezáková et al., 2006], which often requires expensive computations of high dimensional integrals over all possible node arrival sequences. In contrast, we use the likelihood in a much more direct way to evaluate and compare different modeling choices at the level of individual edge placements.

4.3 Preliminaries

Next, we briefly introduce the datasets we use in this chapter, the notation and the experimental methodology we adopt.

4.3.1 Datasets

For each of our four large network datasets, we know the exact time of all the node/edge arrivals. Table 4.1 gives the basic statistics of the four networks. All the networks slowly densify with a densification exponent [Leskovec et al., 2007b] $\alpha \approx 1.2$. All the networks, except DELICIOUS, have shrinking diameter. In FLICKR, ANSWERS, and LINKEDIN, the effective diameter reaches the maximum value of 10 when the network has around 50,000 nodes, and then slowly decreases to the around 7.5; in DELICIOUS, the diameter is practically constant. Also, in all the networks, a majority of edges are bidirectional (column E_b).

Network	FLICKR	DELICIOUS	ANSWERS	LINKEDIN
Time span	03/2003–09/2005	05/2006–02/2007	03/2007–06/2007	05/2003–10/2006
T	621	292	121	1294
N	584,207	203,234	598,314	7,550,955
E	3,554,130	430,707	1,834,217	30,682,028
E_b	2,594,078	348,437	1,067,021	30,682,028
E_u	2,257,211	348,437	1,300,698	30,682,028
E_Δ	1,475,345	96,387	303,858	15,201,596
%	65.63	27.66	23.36	49.55
a	1.32	1.15	1.25	1.14
κ	1.45	0.80	0.95	1.04

Table 4.1: Network dataset statistics. E_b is the number of bidirectional edges, E_u is the number of edges in undirected network, E_Δ is the number of edges that close triangles, % is the fraction of triangle-closing edges, a is the densification exponent ($E(t) \propto N(t)^a$), and κ is the decay exponent ($E_h \propto \exp(-\kappa h)$) of the number of edges E_h closing h hop paths (see Section 4.5 and Figure 4.4).

The reciprocity is 73% in FLICKR, 81% in DELICIOUS, and 58% in ANSWERS; LINKEDIN is undirected, but we know the edge initiator. The fraction of nodes that belongs to the largest weakly connected component is 69% in FLICKR, 72% in DELICIOUS, 81% in ANSWERS, and 91% in LINKEDIN. See Table A.2 for additional information and statistics of these networks.

We consider all networks as undirected but as the edges appear we distinguish between the edge initiator and the edge target. For example, even though edges in LINKEDIN are undirected, the edge initiator is the person that sent the link invitation, and edge target is the node that accepted the invitation.

4.3.2 Notation

Let N , E , and T denote the total number of nodes, edges, and the span of the data in days. Let G_t be a network composed from the earliest t edges, e_1, \dots, e_t for $t \in \{1, \dots, E\}$. Let $t(e)$ be the time when the edge e is created, let $t(u)$ be the time when the node u joined the network, and let $t_k(u)$ be the time when the k^{th} edge of the node u is created. Then $a_t(u) = t - t(u)$ denotes the age of the node u at time t . Let $d_t(u)$ denote the degree of the node u at time t and $d(u) = d_T(u)$. We use $[.]$ to denote a predicate (takes value of 1 if expression is true, else 0). Table 4.1 gives the rest of the symbols.

4.3.3 Maximum-likelihood principle

The maximum-likelihood estimation (MLE) principle can be applied to compare a family of parameterized models in terms of their likelihood of generating the observed data, and as a result, pick the “best” model (and parameters) to explain the data. To apply the likelihood principle, we consider the following setting: we evolve the network edge by edge, and for every edge that arrives into the network, we measure the likelihood that the particular edge endpoints would be chosen under some model. The product of these likelihoods over all edges will give the likelihood of the model. A higher likelihood means a “better” model in the sense that it offers a more likely explanation of the observed data. For numerical purposes, we use log-likelihoods.

SYMBOL	DESCRIPTION
G_t	Graph composed of nodes and edges that arrived before time t
T	Time span of a graph
N	Number of nodes in a graph
E	Number of edges in a graph
$N(t)$	Number of nodes in a graph at time t
$N(e)$	Number of nodes in a graph at time t
e_t	t^{th} edge in a graph
$t(e)$	Time of creation of edge e
$t(u)$	Time when node u joined the network (created its first edge)
$t_i(u)$	Time of creation of i^{th} edge of node u
$a_t(u)$	Age of a node u at time t , $a_t(u) = t - t(u)$
$d(u)$	Final degree of node u
$d_t(u)$	Degree of node u at time t
γ	Power law degree exponent, $p(d) \propto d^{-\gamma}$
a	Densification power law exponent, $E(t) \propto N(t)^a$
$h(u, v)$	Length of the shortest path between nodes u and v
h	Number of hops, path length, distance
E_h	Number of edges that at the time of creation span h hop path
κ	Decay exponent in $E_h \propto \exp(-\kappa h)$
$p_e(d)$	Probability of new edge linking to node of degree d
$p_l(a)$	Node lifetime distribution, <i>i.e.</i> , prob. of node being alive at age a
λ	Node lifetime distribution parameter (exponential distribution)
$\delta_u(d)$	Edge gap, time between d^{th} and $d + 1^{th}$ edge of u , $\delta_u(d) = t_{d+1}(u) - t_d(u)$
α	Power law parameter of edge gap distribution
β	Exponential parameter of edge gap distribution

Table 4.2: Table of symbols.

4.4 Preferential attachment

In this section we study the bias in selection of an edge's source and destination based on the degree and age of the node.

4.4.1 Edge attachment by degree

The preferential attachment (PA) model [Barabási and Albert, 1999] postulates that when a new node joins the network, it creates a constant number of edges, where the destination node of each edge is chosen proportional to the destination's degree. Using our data, we compute the probability $p_e(d)$ that a new edge chooses a destination node of degree d ; $p_e(d)$ is normalized by the number of nodes of degree d that exist just before this step. We compute:

$$p_e(d) = \frac{\sum_t [e_t = (u, v) \wedge d_{t-1}(v) = d]}{\sum_t |\{u : d_{t-1}(u) = d\}|}.$$

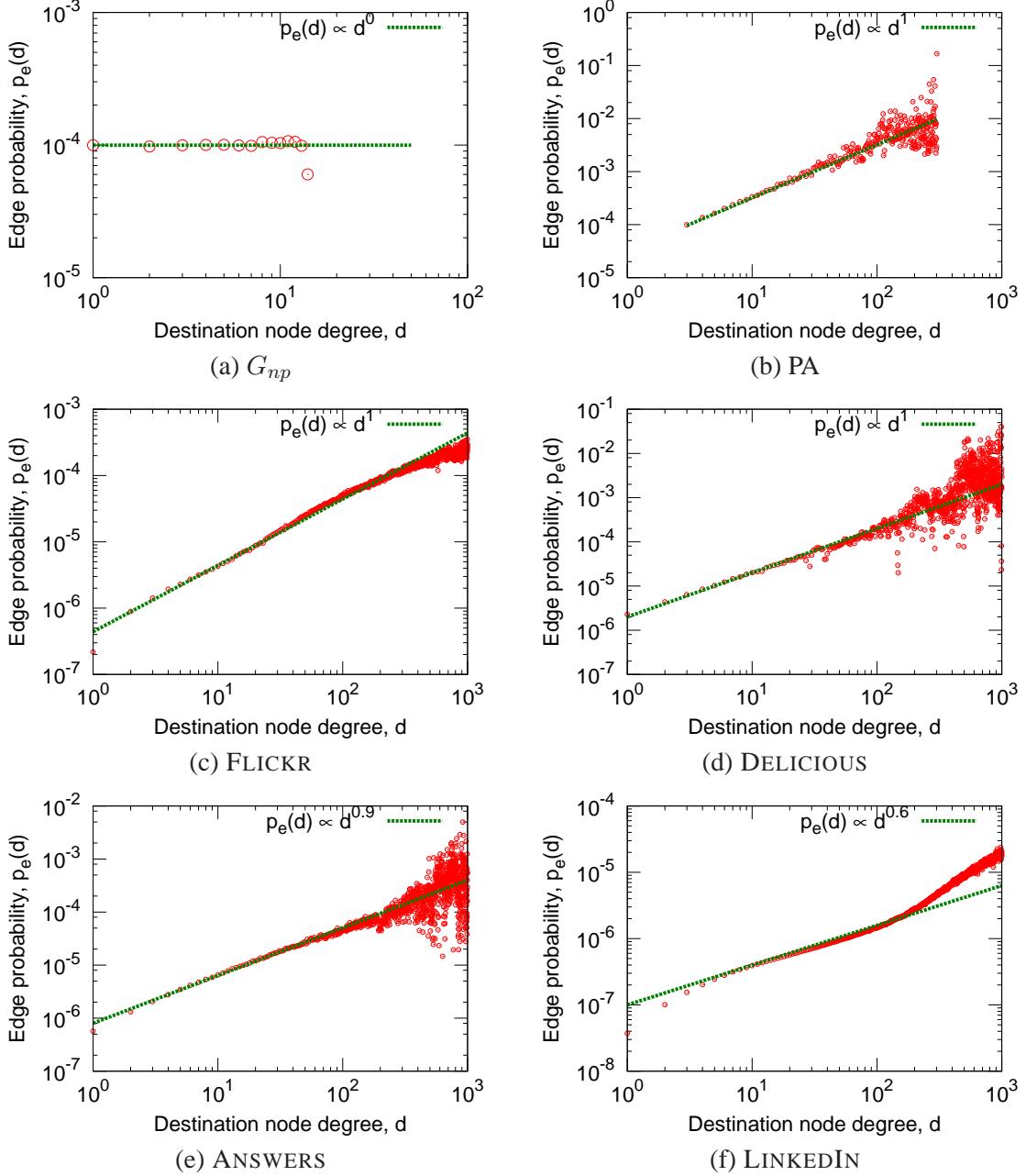


Figure 4.1: Probability $p_e(d)$ of a new edge e choosing a destination at a node of degree d .

First, Figure 4.1(a) shows $p_e(d)$ for the Erdős–Rényi [Erdős and Rényi, 1960] random network, G_{np} , with $p = 12/n$. In G_{np} , since the destination node is chosen independently of its degree, the line is flat. Similarly, in the PA model, where nodes are chosen proportionally to their degree, we get a linear relationship $p_e(d) \propto d$; see Figure 4.1(b).

Next we turn to our four networks and fit the function $p_e(d) \propto d^\tau$. In FLICKR, Figure 4.1(c), degree 1 nodes have lower probability of being linked as in the PA model; the rest of the edges could be explained well by PA. In DELICIOUS, Figure 4.1(d), the fit nicely follows PA. In ANSWERS, Figure 4.1(e), the

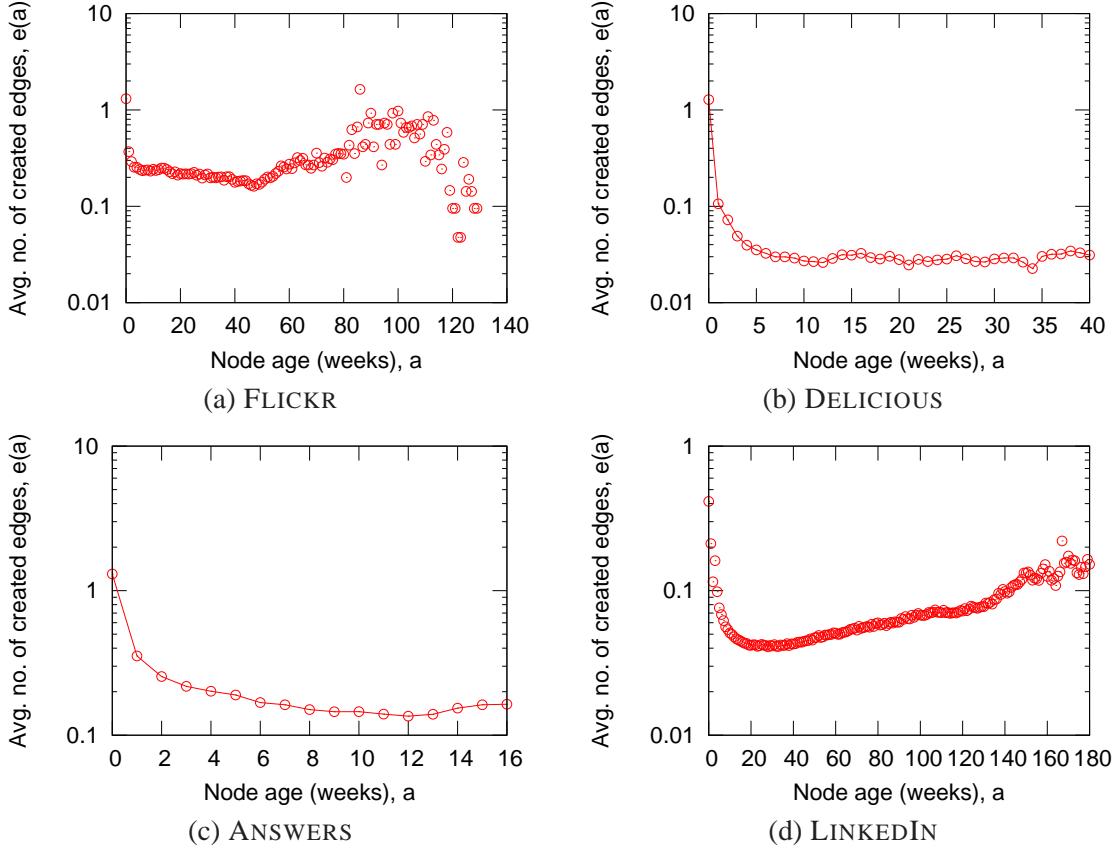


Figure 4.2: Average number of edges created by a node of age a .

presence of PA is slightly weaker, with $p_e(d) \propto d^{0.9}$. LINKEDIN has a very different pattern: edges to the low degree nodes do not attach preferentially (the fit is $d^{0.6}$), whereas edges to higher degree nodes are more “sticky” (the fit is $d^{1.2}$). This suggests that high-degree nodes in LINKEDIN get super-preferential treatment.

To summarize, even though there are minor differences in the exponents τ for each of the four networks, we can treat $\tau \approx 1$, meaning, the attachment is essentially linear. This observation is a bit different from what was observed by Capocci *et al.* [Capocci et al., 2006] who observed the (sublinear, $\tau = 0.9$) preferential attachment up to page degree $d \approx 100$ and for $d > 100$ linking probability actually *decreased* with node degree d .

4.4.2 Edges by the age of the node

Next, we examine the effect of a node’s age on the number of edges it creates. The hypothesis is that older, more experienced users are also more engaged and thus create more edges.

Figure 4.2 plots the fraction of edges initiated by nodes of a certain age. Then $e(a)$, the average number of edges created by nodes of age a , is the number of edges created by nodes of age a normalized by the number of nodes that achieved age a :

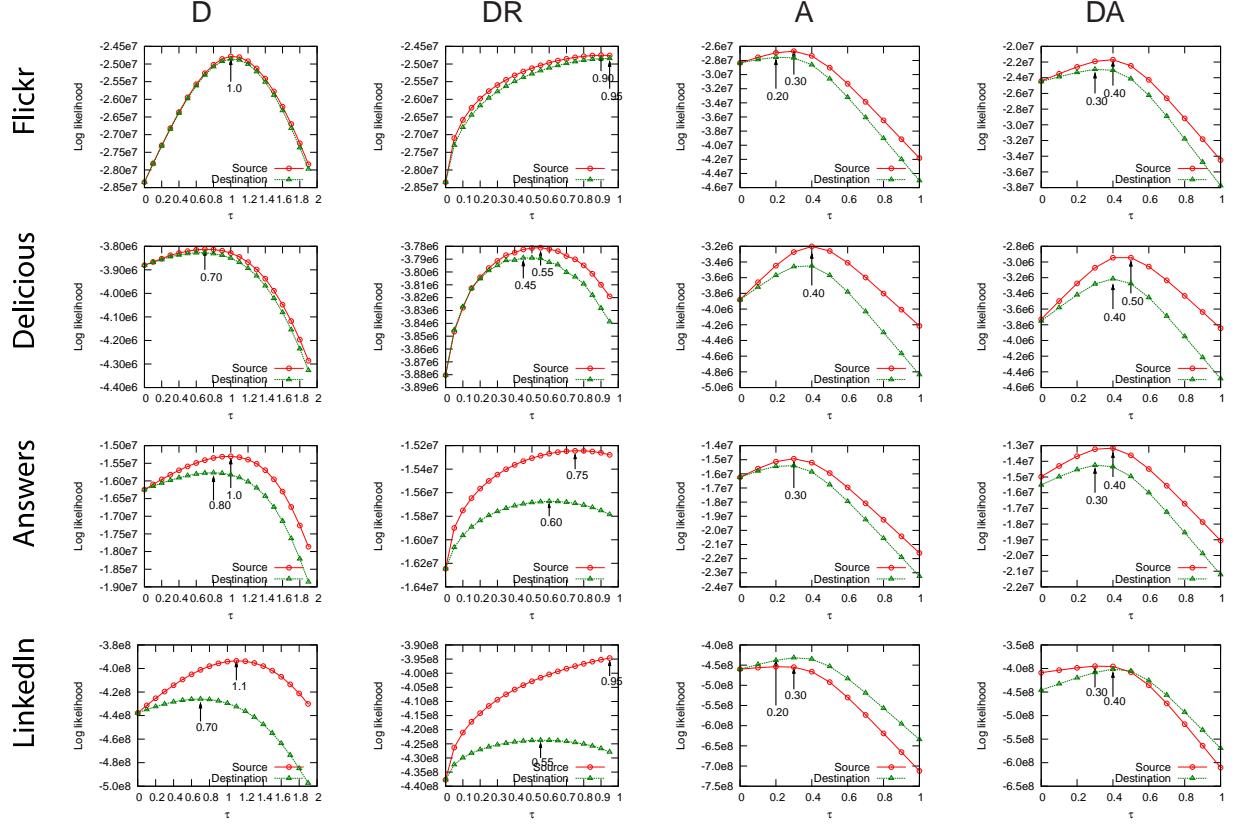


Figure 4.3: Log-likelihood of an edge selecting its source and destination node. Arrows denote τ at highest likelihood.

$$e(a) = \frac{|\{e = (u, v) : t(e) - t(u) = a\}|}{|\{t(u) : t_\ell - t(u) \geq a\}|},$$

where t_ℓ is the time when the last node in the network joined.

Notice a spike at nodes of age 0. These correspond to the people who receive an invite to join the network, create a first edge, and then never come back. Typically these are the users who are not yet part of the social network service, they receive an invitation to join as one of the existing members invited them. By accepting the invitation and registering they also create a link but never come back to use the service again. For all other ages, the level of activity seems to be uniform over time, except for LINKEDIN, in which activity of older nodes slowly increases over time.

4.4.3 Bias towards node age and degree

Using the MLE principle, we study the combined effect of node age and degree by considering the following four parameterized models for choosing the edge endpoints at time t .

- D: The probability of selecting a node v is proportional to its current degree raised to power τ : $d_t(v)^\tau$.

- DR: With probability τ , the node v is selected preferentially (proportionally to its degree), and with probability $(1 - \tau)$, uniformly at random: $\tau \cdot d_t(v) + (1 - \tau) \cdot 1/N(t)$.
- A: The probability of selecting a node is proportional to its age raised to power τ : $a_t(v)^\tau$
- DA: The probability of selecting a node v is proportional to the product of its current degree and its age raised to the power τ : $d_t(v) \cdot a_t(v)^\tau$.

The experiment goes as follows. We unroll the evolution of the network edge by edge. Then for each edge e_t we take current state of the graph G_{t-1} at time $t - 1$ and we consider the probability of selecting the source and destination node of e_t under one of the above four models and fixed τ . We repeat this for each value of τ and plot the log-likelihood separately for selection of edge source and edge destination node.

Figure 4.3 plots the log-likelihoods under different models, as a function of τ . The red curve plots the log-likelihood of selecting a source node and the green curve for selecting the destination node of an edge.

In FLICKR the selection of destination is purely preferential: model D achieves the maximum likelihood at $\tau = 1$, and model DA is very biased to model D, *i.e.*, $\tau \approx 1$. Model A has worse likelihood but model DA improves the overall log-likelihood by around 10%. Edge attachment in DELICIOUS seems to be the most “random”: model D has worse likelihood than model DR. Moreover the likelihood of model DR achieves maximum at $\tau = 0.5$ suggesting that about 50% of the DELICIOUS edges attach randomly. Model A has better likelihood than the degree-based models, showing edges are highly biased towards young nodes. For ANSWERS, models D, A, and DR have roughly equal likelihoods (at the optimal choice of τ), while model DA further improves the log-likelihood by 20%, showing some age bias. In LINKEDIN, age-biased models are worse than degree-biased models. We also note strong degree preferential bias of the edges. As in FLICKR, model DA improves the log-likelihood by 10%.

We notice that selecting an edge’s destination node is harder than selecting its source (the green curve is usually below the red). Also, selecting a destination appears more random than selecting a source — the maximum likelihood τ of the destination node (green curve) for models D and DR is shifted to the left when compared to the source node (red), which means the degree bias is weaker. Similarly, there is a stronger bias towards young nodes in selecting an edge’s source than in selecting its destination. Based on the observations, we conclude that PA (model D) performs reasonably well compared to more sophisticated variants based on degree and age.

4.5 Locality of edge attachment

Even though our analysis suggests that PA is a reasonable model for edge destination selection, it is inherently “non-local” in that edges are no more likely to form between nodes which already have friends in common. In this section we perform a detailed study of the locality properties of edge destination selection.

We first consider the following notion of **edge locality**: for each new edge (u, w) , we measure the number of hops it spans, *i.e.*, the length of the shortest path between nodes u and w immediately before the edge was created. In Figure 4.4 we study the distribution of these shortest path values induced by each new edge for G_{np} (with $p = 12/n$), PA, and the four social networks. (The isolated dot on the left counts the number of edges that connected previously disconnected components of the network.)

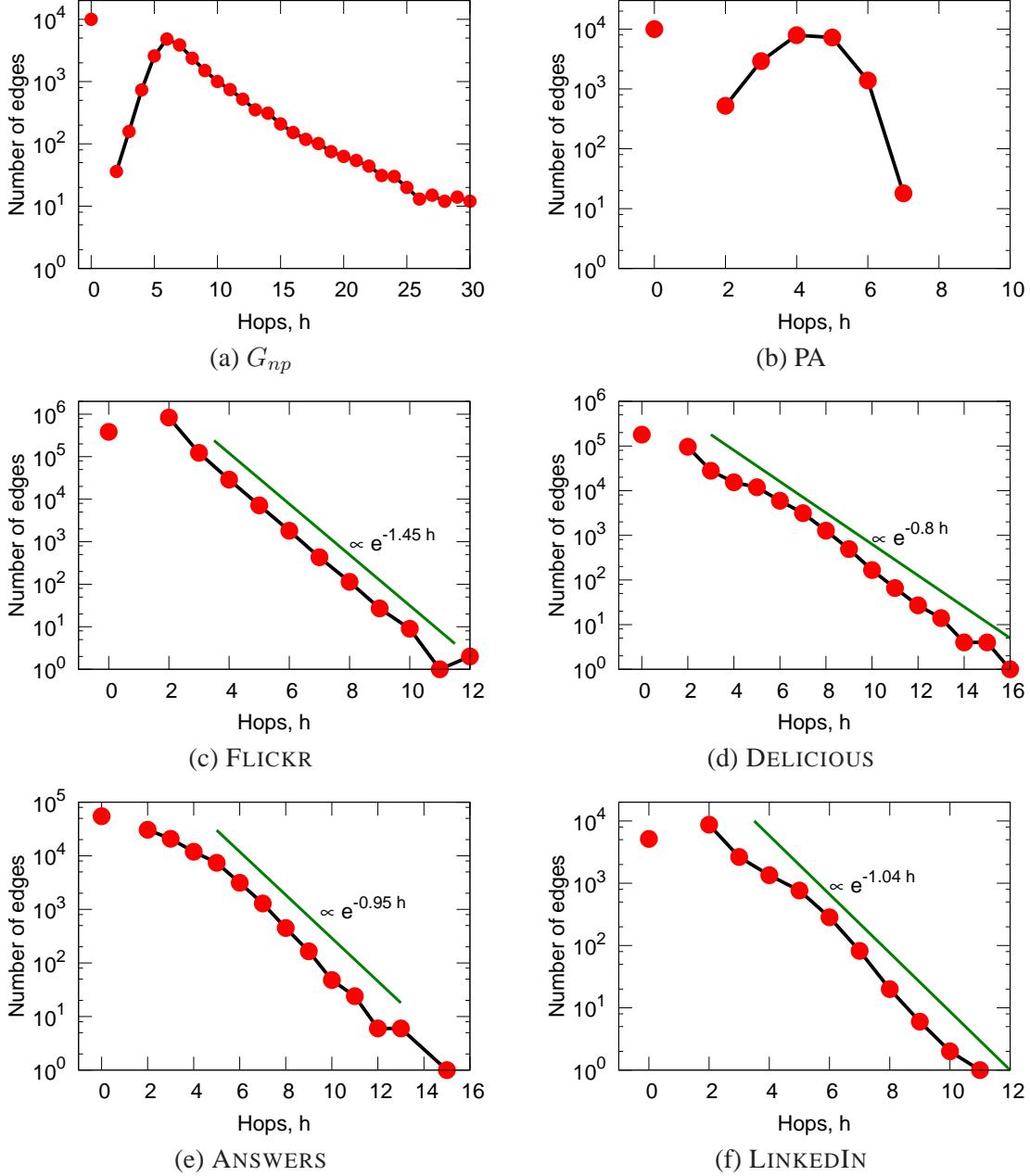


Figure 4.4: Number of edges E_h created to nodes h hops away. $h = 0$ counts the number of edges that connected previously disconnected components.

For G_{np} most new edges span nodes that were originally six hops away, and then the number decays polynomially in the hops. In the PA model, we see a lot of long-range edges; most of them span four hops but none spans more than seven. The hop distributions corresponding to the four real-world networks look similar to one another, and strikingly different from both G_{np} and PA. The number of edges decays exponentially with the distance between the nodes (see Table 4.1 for fitted decay exponents κ). This means that most edges are created between nodes that are close. The exponential decay suggests that the creation

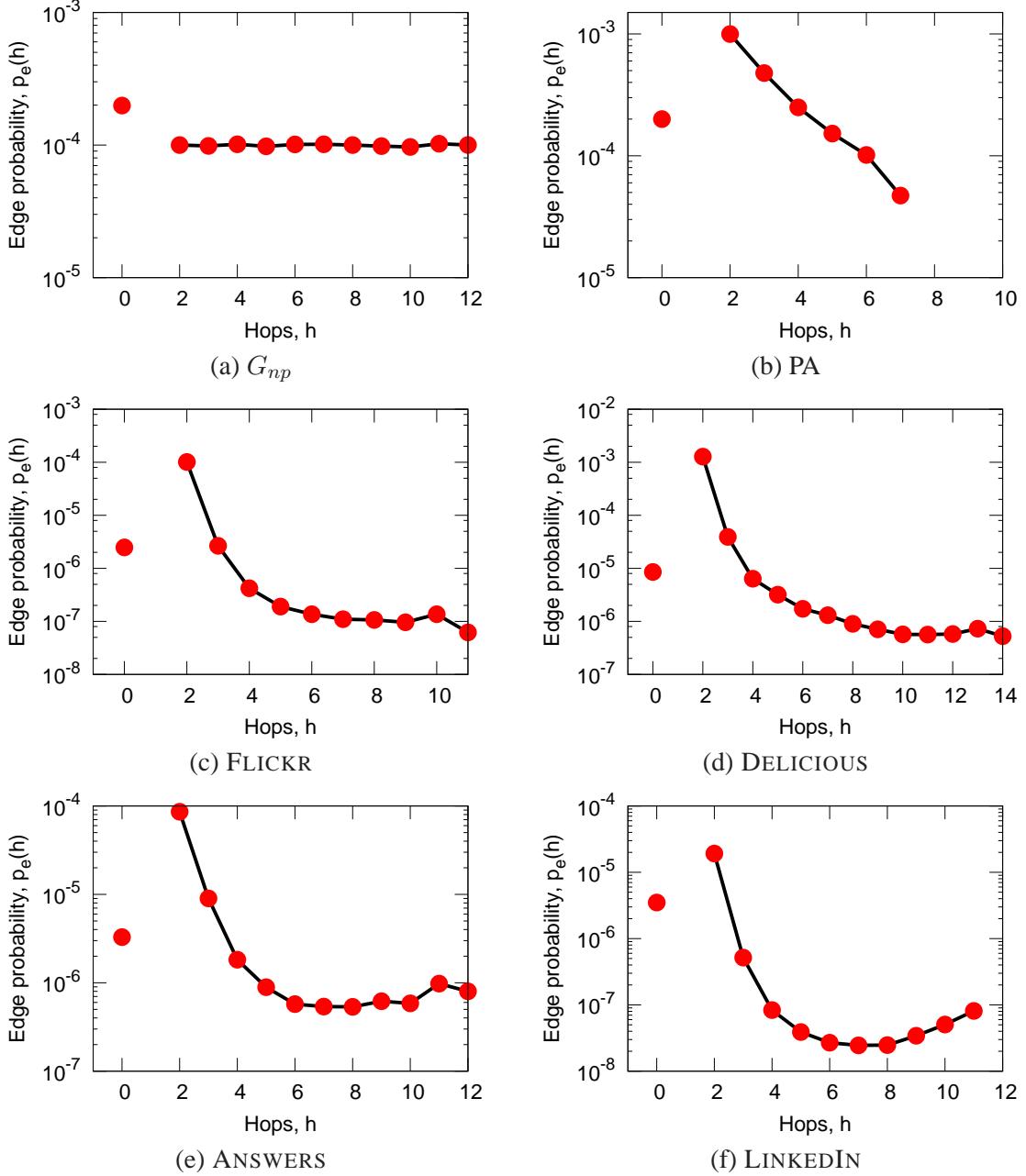


Figure 4.5: Probability of linking to a random node at h hops from source node. Value at $h = 0$ hops is for edges that connect previously disconnected components.

of a large fraction of edges can be attributed to locality in the network structure, namely most of the times people who are close in the network (*e.g.*, have a common friend) become friends themselves.

These results involve counting the number of edges that link nodes certain distance away. In a sense, this overcounts edges (u, w) for which u and w are far away, as there are many more distant candidates to choose from — it appears that the number of long-range edges decays exponentially while the number of

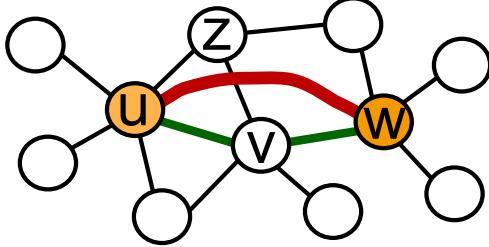


Figure 4.6: Triangle-closing model: node u creates an edge by selecting intermediate node v , which then selects target node w to which the edge (u, w) is created.

long-range candidates grows exponentially. To explore this phenomenon, we count the number of hops each new edge spans but then normalize the count by the total number of nodes at h hops. More precisely, we compute

$$p_e(h) = \frac{\sum_t [e_t \text{ connects nodes at distance } h \text{ in } G_{t-1}]}{\sum_t (\# \text{ nodes at distance } h \text{ from the source node of } e_t)}.$$

First, Figures 4.5(a) and (b) show the results for G_{np} and PA models. (Again, the isolated dot at $h = 0$ plots the probability of a new edge connecting disconnected components.) In G_{np} , edges are created uniformly at random, and so the probability of linking is independent of the number of hops h between the edge endpoints and thus $p_l(h)$ is flat. In PA, due to degree correlations short (local) edges prevail. However, a non-trivial amount of probability goes to edges that span more than two hops. (Notice the logarithmic y -axis.)

Figures 4.5(c)–(f) show the plots for the four networks. The probability of linking to a node h hops away decays very quickly, seemingly double-exponentially, *i.e.*, $p_e(h) \propto \exp(\exp(-h))$ (fits not shown). This behavior is drastically different from both the PA and G_{np} models. Also note that almost all of the probability mass is on edges that close length-two paths. This means that edges are most likely to close triangles, *i.e.*, connect people with common friends.

Column E_Δ in Table 4.1 further illustrates this point by presenting the number of triangle-closing edges. FLICKR and LINKEDIN have the highest fraction of triangle-closing edges, whereas ANSWERS and DELICIOUS have substantially less such edges. Note that here we are not measuring the fraction of nodes participating in triangles. Rather, we unroll the evolution of the network, and for every new edge check to see if it closes a new triangle or not.

4.5.1 Triangle-closing models

Given that such a high fraction of edges close triangles, we aim to model how a length-two path should be selected. We consider a scenario in which a source node u has decided to add an edge to some node w two hops away, and we are faced with various alternatives for the choice of node w . Figure 4.6 illustrates the setting. Edges arrive one by one and the simplest model to close a triangle (edge (u, w) in the figure) is to have u select a destination w randomly from all nodes at two hops from u .

To improve upon this baseline model we consider various models of choosing node w . We consider processes in which u first selects a neighbor v according to some mechanism, and v then selects a neighbor

w according to some (possibly different) mechanism. The edge (u, w) is then created and the triangle (u, v, w) is closed. The selection of both v and w involves picking a neighbor of a node.

We consider five different models of choosing a neighbor v of u . Node v is chosen:

- **random**: uniformly at random,
- **deg $^\tau$** : proportional to degree raised to power τ , $d(v)^\tau$,
- **com**: prop. to the number of common friends $c(u, v)$ with u ,
- **last $^\tau$** : proportional to the time passed since v last created an edge raised to power τ ,
- **comlast $^\tau$** : proportional to the product of the number of common friends with u and the last activity time, raised to power τ .

As stated before, we can compose any two of these basic models to choose a two-hop neighbor, *i.e.*, a way to close the triangle. For instance, the **last $^{0.1}$ -com** model will work as follows: u will employ the **last $^{0.1}$** model to select node v , v will then employ the **com** model to select node w , and then u will add an edge to w , closing the triangle (u, v, w) . We consider all 25 five possible composite models for selecting a two-hop neighbor and evaluate them by the likelihood that the model generated all the edges that closed length-two paths in the real network.

Table 4.3 shows the percent improvement of various triangle-closing models over the log-likelihood of choosing a two-hop neighbor uniformly at random as a destination of the edge (the baseline). The simplest model, **random-random**, works remarkably well. Initially, we were somewhat surprised by this. However, if one thinks about the **random-random** it has many desirable properties. For example, it gives higher probability to nodes with more length-two paths, discounting each path by roughly $1/d(v)$. Moreover, it is also biased towards high-degree nodes, as they have multiple paths leading towards them.

The **deg $^{1.0}$ -random** model weighs each node w by roughly the number of length-two paths between u and w . However, we find that it performs worse than **random-random**. For the more general **deg $^\tau$ -random**, the optimal value of τ varies from 0.1 to 0.3 over all the four networks, and this model provides meaningful improvements only for the ANSWERS network.

The **com** model considers the strength of a tie between u and v , which we approximate by the number of common friends $c(u, v)$ of nodes u and v ; the larger the value, the stronger the tie. By selecting v with probability proportional to $c(u, v)$, we get a substantial gain in model likelihood. A factor that further improves the model is the recency of activity by v , captured by **last $^\tau$** . By selecting nodes that have recently participated in a new edge with higher probability, we get another sizable improvement in the model likelihood. These two capture the finer details of network evolution.

In summary, while degree helps marginally, for all the networks, the **random-random** model gives a sizable chunk of the performance gain over the baseline (10%). Due its simplicity, we choose this as the triangle-closing model for the rest of the chapter.

Note that the above methodology could be extended to edge creations other than triangle-closing. We chose to focus on the triangle-closing edges for two reasons. First, a high fraction of all edges created fall into this category, and hence an understanding of triangle-closing edges is an important first step towards understanding the overall network evolution. Second, with the exception of quite simplistic models, it is computationally infeasible to compute the likelihood at a distance greater than two hops as the number of nodes and possible paths increases dramatically.

FLICKR	random	$\text{deg}^{0.2}$	com	$\text{last}^{-0.4}$	$\text{comlast}^{-0.4}$
random	13.6	13.9	14.3	16.1	15.7
$\text{deg}^{0.1}$	13.5	14.2	13.7	16.0	15.6
$\text{last}^{0.2}$	14.7	15.6	15.0	17.2	16.9
com	11.2	11.6	11.9	13.9	13.4
$\text{comlast}^{0.1}$	11.0	11.4	11.7	13.6	13.2
DELICIOUS	random	$\text{deg}^{0.3}$	com	$\text{last}^{-0.2}$	$\text{comlast}^{-0.2}$
random	11.7	12.4	13.8	13.2	15.1
$\text{deg}^{0.2}$	12.2	12.8	14.3	13.7	15.6
$\text{last}^{-0.3}$	13.8	14.6	16.0	15.3	17.2
com	13.6	14.4	15.8	15.2	17.1
$\text{comlast}^{-0.2}$	14.7	15.6	16.9	16.3	18.2
ANSWERS	random	$\text{deg}^{0.3}$	com	$\text{last}^{-0.2}$	$\text{comlast}^{-0.2}$
random	6.80	10.1	11.8	9.70	13.3
$\text{deg}^{0.2}$	7.18	10.5	12.2	10.1	13.7
$\text{last}^{-0.3}$	9.95	13.4	15.0	12.8	16.4
com	6.82	10.3	11.8	9.80	13.4
$\text{comlast}^{0.2}$	7.93	11.5	12.9	10.9	14.5
LINKEDIN	random	$\text{deg}^{0.1}$	com	$\text{last}^{-0.1}$	$\text{comlast}^{-0.1}$
random	16.0	16.5	18.2	17.2	18.5
$\text{deg}^{0.1}$	15.9	16.4	18.0	17.0	18.4
$\text{last}^{-0.1}$	19.0	19.5	21.1	20.0	21.4

Table 4.3: Triangle-closing models. First pick intermediate node v (fix column), then target node w (fix row). The cell gives percent improvement over the log-likelihood of picking a random node two hops away (baseline).

4.6 Node and edge arrival process

In this section we turn our focus to the edge initiation process that determines which node is responsible for creating a new edge (Section 4.6.1), and then to the process by which new nodes arrive into the network (Section 4.6.2).

4.6.1 Edge initiation

In the following we assume that the sequence and timing of node arrivals is given, and we model the process by which nodes initiate edges. We begin by studying how long a node remains active in the social network, and then during this active lifetime, we study the specific times at which the node initiates new edges.

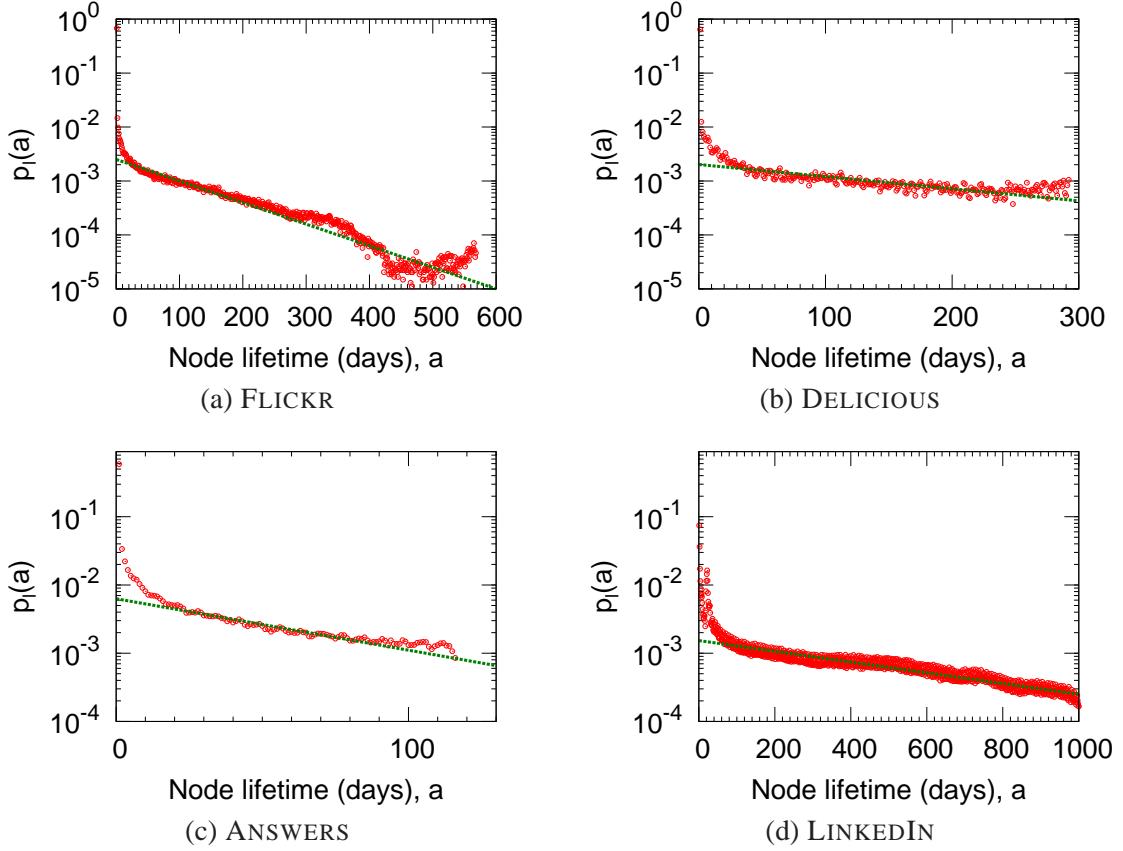


Figure 4.7: Exponentially distributed node lifetimes.

Node lifetime

To avoid truncation effects, we only consider those nodes whose last-created edge is in the first half of all edges in the data. Recall that the lifetime of a node u is $a(u) = t_{d(u)}(u) - t_1(u)$. We evaluate the likelihood of various distributions and observe that node lifetimes are best modeled by an exponential distribution, $p_\ell(a) = \lambda \exp(-\lambda a)$. Figure 4.7 gives the plot of the data and the exponential fits, where time is measured in days. In Table 4.6, the row corresponding to λ gives the values of fitted exponents. We note that the exponential distribution does not fit well the nodes with very short lifetimes, *i.e.*, nodes that are invited into the network, create an edge and never return. But the distribution provides a very clean fit for nodes whose lifetime is more than a week.

Time gap between the edges

Now that we have a model for the lifetime of a node u , we must model that amount of elapsed time between edge initiations from u . Let $\delta_u(d) = t_{d+1}(u) - t_d(u)$ be the time it takes for the node u with current degree d to create its $(d + 1)$ -st out-edge; we call $\delta_u(d)$ the *edge gap*. Again, we examine several candidate distributions to model edge gaps. Table 4.4 shows the percent improvement of the log-likelihood at the MLE over the exponential distribution. The best likelihood is provided by a power law with exponential

degree d	power law	power law exp. cutoff	log normal	stretched exp.
1	9.84	12.50	11.65	12.10
2	11.55	13.85	13.02	13.40
3	10.53	13.00	12.15	12.59
4	9.82	12.40	11.55	12.05
5	8.87	11.62	10.77	11.28
avg., $d \leq 20$	8.27	11.12	10.23	10.76

Table 4.4: Edge gap distribution: percent improvement of the log-likelihood at MLE over the exponential distribution.

cutoff: $p_g(\delta(d); \alpha, \beta) \propto \delta(d)^{-\alpha} \exp(-\beta\delta(d))$, where d is the current degree of the node. (Note that the distribution is neither exponential nor Poisson, as one might be tempted to assume.) We confirm these results in Figure 4.8, in which we plot the MLE estimates to gap distribution $\delta(1)$, *i.e.*, distribution of times that it took a node of degree 1 to add the second edge. In fact, we find that all gaps distributions $\delta(d)$ are best modeled by a power law with exponential cut-off (Table 4.4 gives improvements in log-likelihoods for $d = 1, \dots, 5$ and the average for $d = 1, \dots, 20$.) The hump in LINKEDIN dataset can be explained by external event and the way the LinkedIn service operates.

For each $\delta(d)$ we fit a separate distribution and Figure 4.9 shows the evolution of the parameters α and β of the gap distribution, as a function of the degree d of the node. Interestingly, the power law exponent $\alpha(d)$ remains *constant* as a function of d , at almost the same value for all four networks. On the other hand, the exponential cutoff parameter $\beta(d)$ increases *linearly* with d , and varies by an order of magnitude across networks; this variation models the extent to which the “rich get richer” phenomenon manifests in each network. This means that the slope α of power law part remains constant, only the exponential cutoff part (parameter β) starts to kick in sooner and sooner. So, nodes add their $(d + 1)^{st}$ edge faster than their d^{th} edge, *i.e.*, nodes start to create more and more edges (sleeping times get shorter) as they get older (and have higher degree). So, based on Figure 4.9, the overall gap distribution can be modeled by the power law with exponential cutoff distribution where the exponential cutoff parameter β increases linearly with current node degree d : $p_g(\delta|d; \alpha, \beta) \propto \delta^{-\alpha} \exp(-\beta d \delta)$.

This is interesting finding as it very accurately models node dynamics. Nodes sleep, wake up, create edges and go back to sleep. As nodes get older they keep adding edges faster. However, the power-law slope of gap time distribution remains constant with node degree. But it is the exponential cutoff parameter that starts getting stronger and stronger and cuts the tail of the power law part, which makes the sleeping times shorter and shorter.

Given the above observation, a natural hypothesis would be that nodes that will attain high degree in the network are in some way *a priori* special, *i.e.*, they correspond to “more social” people who would inherently tend to have shorter gap times and enthusiastically invite friends at a higher rate than others, attaining high degree quickly due to their increased activity level. However, this phenomenon does not occur in any of the networks. We computed the correlation coefficient between $\delta(1)$ and the final degree $d(u)$ of a node u . The correlation values are -0.069 for DELICIOUS, -0.043 for FLICKR, -0.036 for ANSWERS, and -0.027 for LINKEDIN. Thus, there is almost no correlation, which shows that the gap distribution is independent of a node’s final degree. It only depends on node lifetime, *i.e.*, high degree nodes are not *a priori* special, they just live longer, and accumulate many edges.

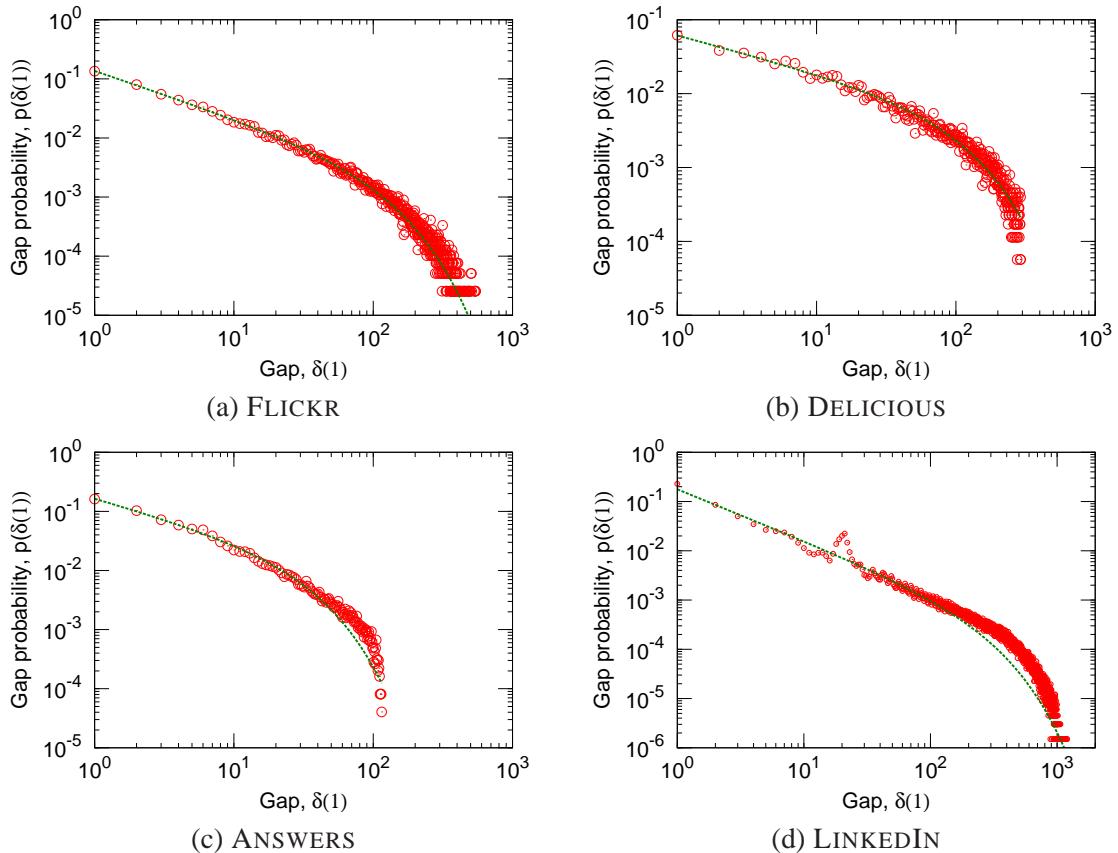


Figure 4.8: Edge gap distribution for a node to obtain the second edge, $\delta(1)$, and MLE power law with exponential cutoff fits.

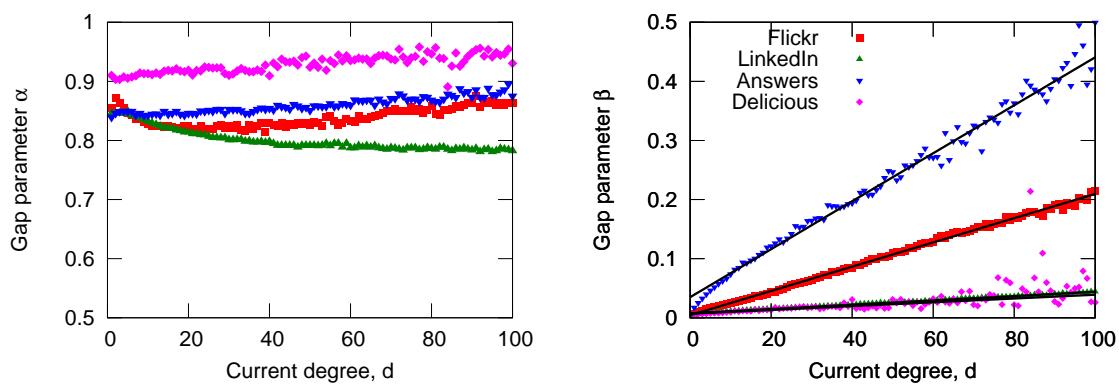


Figure 4.9: Evolution of the α and β parameters with the current node degree d . α remains constant, and β linearly increases.

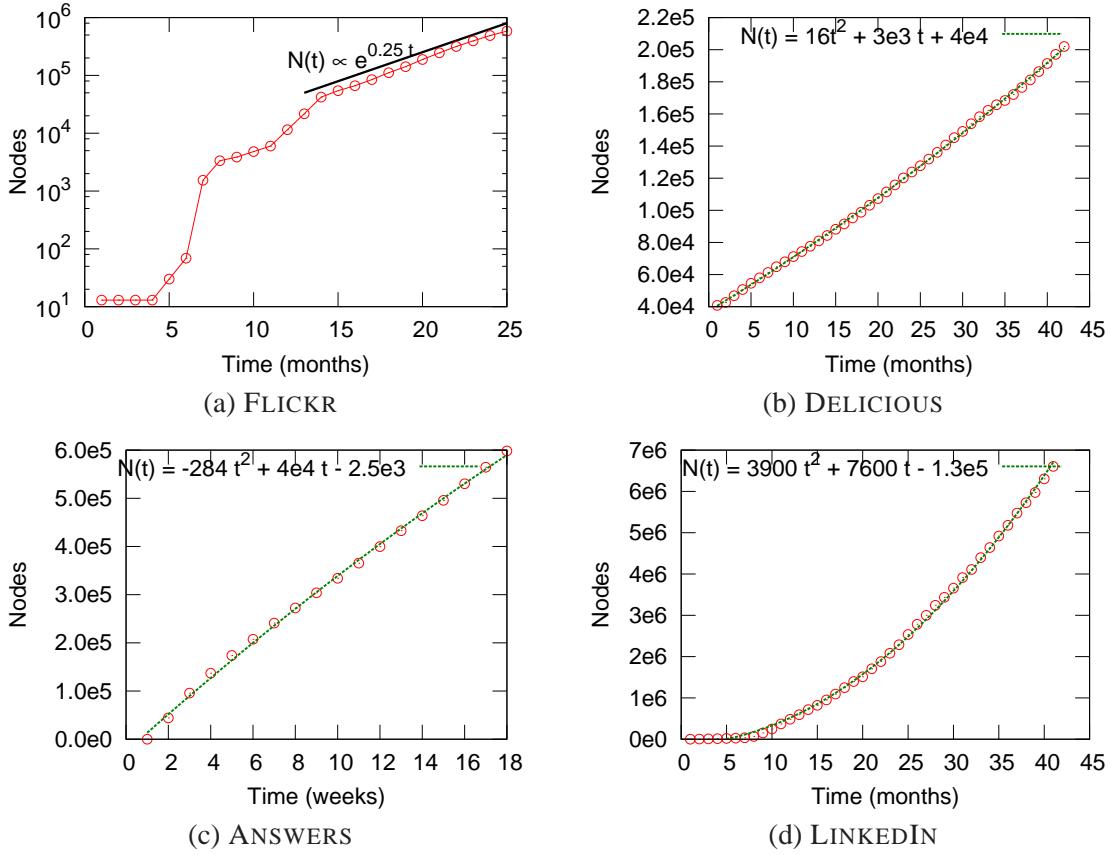


Figure 4.10: Number of nodes over time.

Network	$N(t)$
FLICKR	$\exp(0.25t)$
DELICIOUS	$16t^2 + 3000t + 40000$
ANSWERS	$-284t^2 + 40000t - 2500$
LINKEDIN	$3900t^2 + 7600t - 130000$

Table 4.5: Node arrival functions for the four network datasets. Figure 4.10 plots the number of nodes over time.

4.6.2 Node arrivals

Finally, we turn to the question of modeling node arrivals into the system. Figure 4.10 shows the number of users in each of our networks over time, and Table 4.5 captures the best fits. FLICKR grows exponentially over much of our network, while the growth of other networks is much slower. DELICIOUS grows slightly superlinearly, LINKEDIN quadratically, and ANSWERS sublinearly. Given these wild variations we conclude the node arrival process needs to be specified in advance as it varies greatly across networks due to external factors.

4.7 A network evolution model

Next we present our network evolution model. In contrast to Preferential Attachment, Copying or Forest Fire model where nodes arrive one at a time, immediately create all their edges and then essentially die, our model describes the evolution much more precisely as in our model nodes appear, create one edge at a time, then go to sleep, wake up, create next edge and so on until they die. So we model complete temporal arrival and creation process of both nodes and edges.

First let's take stock of what we measured and observed so far:

- (a) In Section 4.6.2, we analyzed the node arrival rates and showed that they are network-dependent and can be succinctly represented by a node arrival function $N(t)$ that is either a polynomial or an exponential.
- (b) In Section 4.6.1, we analyzed the node lifetimes and showed they are exponentially distributed with parameter λ .
- (c) In Section 4.4.1, we argued that the destination of the first edge of a node is chosen proportional to its degree (*i.e.*, preferentially attached).
- (d) In Section 4.6.1, we analyzed the time gaps between edge creation at a node and showed they can be captured by a power law with exponential cutoff, with parameters α, β .
- (e) In Section 4.5, we showed that most of the edges span two hops, and the simple random-random triangle-closing model works well.

Motivated by these observations, we now present a complete network evolution model. Our model is parameterized by $N(\cdot), \lambda, \alpha, \beta$, and operates as follows.

1. Nodes arrive using the node arrival function $N(\cdot)$.
2. Node u arrives and samples its lifetime a from the exponential distribution $p_\ell(a) = \lambda \exp(-\lambda a)$.
3. Node u adds the first edge to node v with probability proportional to its degree.
4. A node u with degree d samples a time gap δ from the edge gap distribution $p_g(\delta|d; \alpha, \beta) = (1/Z)\delta^{-\alpha} \exp(-\beta d \delta)$ and goes to sleep for δ time steps.
5. When a node wakes up, if its lifetime has not expired yet, it creates a two-hop edge using the random-random triangle-closing model.
6. If a node's lifetime has expired, then it stops adding edges; otherwise it repeats from step 4.

The values of $N(\cdot)$ for the four networks are given in Table 4.5 and the values of α, β, λ are given in Table 4.6.

Note that one could also use more sophisticated edge destination selection strategies like the random surfer model [Blum et al., 2006] or other triangle-closing techniques as discussed in Section 4.5.1. For example, in step 5, a node u can pick a sequence of nodes ($u = w_0, w_1, \dots, w_k = w$), where each w_i is picked uniformly from the neighbors of w_{i-1} , and the sequence length k is chosen from the distribution in Figure 4.4. Node u then links to w .

4.7.1 Gaps and power law degree distribution

We now show that our model, node lifetime combined with gaps, produces power law out-degree distribution. This is interesting as a model of temporal behavior (lifetime plus gaps) gives rise to a structural network property (*i.e.*, power law out degree distribution).

Theorem 4.7.1. *The out-degrees are distributed according to a power law with exponent*

$$\gamma = 1 + \frac{\lambda\Gamma(2-\alpha)}{\beta\Gamma(1-\alpha)}. \quad (4.1)$$

Proof. We first compute the normalizing constant Z of the gap distribution $p_g(\delta|d; \alpha, \beta)$:

$$Z = \int_0^\infty \delta^{-\alpha} e^{-\beta d \delta} d\delta = \frac{\Gamma(1-\alpha)}{(\beta d)^{1-\alpha}}. \quad (4.2)$$

Let a be the lifetime sampled from the exponential distribution $p_\ell(a) = \lambda \exp(-\lambda a)$. Recall the edge creation process: a node adds its first edge and samples the next gap $\delta(1)$ according to $p_g(\cdot)$, sleeps for $\delta(1)$ time units, creates the second edge, samples a new gap $\delta(2)$ according to $p_g(\cdot)$, sleeps for $\delta(2)$ units, and so on until it uses up all of its lifetime a . This means that for a node u with lifetime $a = a(u)$ and final degree $D = d(u)$, we have

$$\sum_{d=1}^D \delta(k) \leq a. \quad (4.3)$$

Analogous to (4.2), we obtain the expected time gap $E(\delta|d; \alpha, \beta)$ for a node of degree d :

$$E(\delta|d; \alpha, \beta) = \frac{\Gamma(2-\alpha)}{\Gamma(1-\alpha)} (\beta d)^{-1}. \quad (4.4)$$

Combining (4.3) and (4.4), we relate the lifetime a and the expected final degree D of a node:

$$\sum_{d=1}^D \frac{\Gamma(2-\alpha)}{\Gamma(1-\alpha)} (\beta d)^{-1} = \frac{\Gamma(2-\alpha)}{\Gamma(1-\alpha)} \beta^{-1} \sum_{d=1}^D d^{-1} \leq a. \quad (4.5)$$

Notice that $\sum_{d=1}^D d^{-1} = \Theta(\ln D)$. From (4.5), the final degree D of a node with lifetime a is

$$D \approx \exp\left(\frac{\Gamma(1-\alpha)}{\Gamma(2-\alpha)} \beta a\right).$$

Thus, D is an exponential function of the age a , *i.e.*, $D = r(a) = \exp(\mu a)$, where $\mu = \frac{\Gamma(1-\alpha)}{\Gamma(2-\alpha)} \beta$.

Since node lifetimes are exponentially distributed with parameter λ , we now compute the distribution of D as a function of λ and μ as follows:

$$D \sim p_\ell(r^{-1}(D)) \left| \frac{\partial r^{-1}(D)}{\partial D} \right| = \frac{\lambda}{\mu D} e^{-(\lambda/\mu) \log D} = \frac{\lambda D^{-(1+\lambda/\mu)}}{\mu}.$$

Thus, the degree distribution in our gap model follows a power law with exponent $1 + \lambda/\mu$, completing the proof. \square

	FLICKR	DELICIOUS	ANSWERS	LINKEDIN
λ	0.0092	0.0052	0.019	0.0018
α	0.84	0.92	0.85	0.78
β	0.0020	0.00032	0.0038	0.00036
true γ	1.73	2.38	1.90	2.11
predicted γ	1.74	2.30	1.75	2.08

Table 4.6: Predicted by Theorem 4.7.1 vs. true degree exponents.

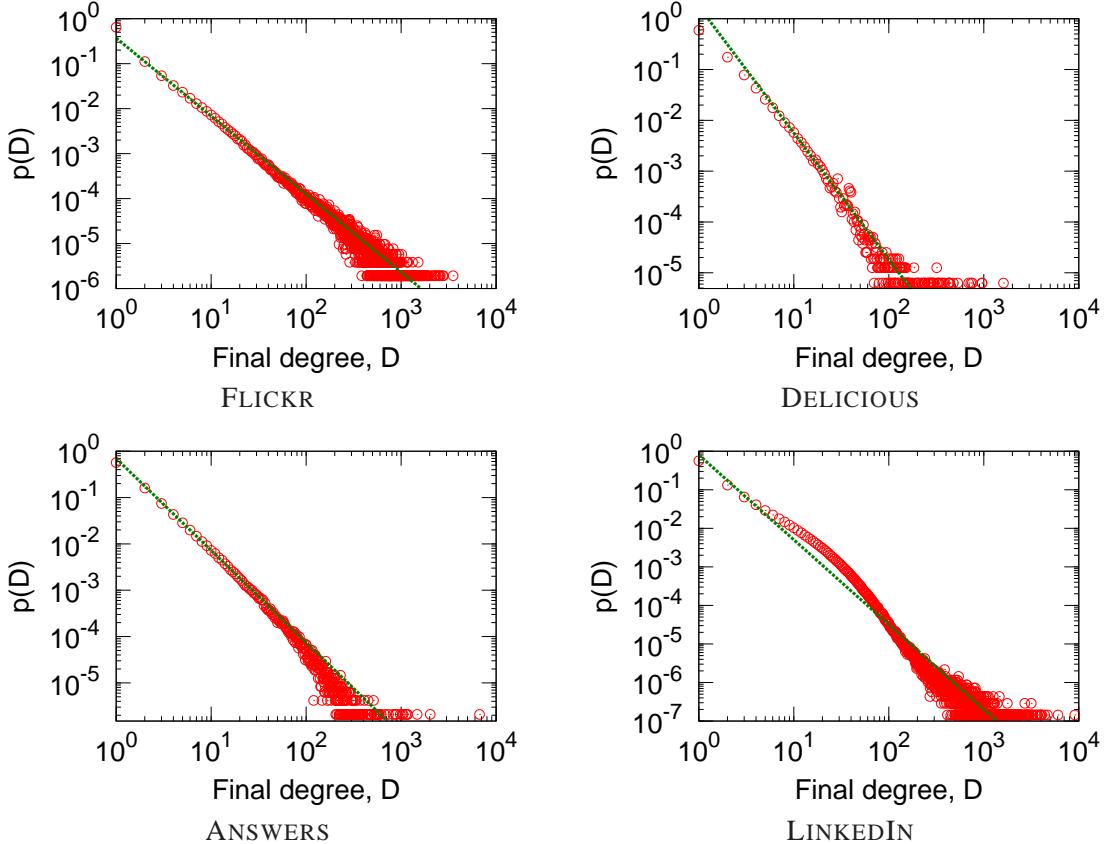


Figure 4.11: Degree distribution and power law fits.

4.7.2 Validation of the model

We validate the accuracy of our modeling assumptions by empirically estimating the lifetime λ , and gap distribution α, β parameter values for each network. We then apply Theorem 4.7.1, which yields the power law degree exponents produced by our model. Then we empirically measure the true power law degree exponents of the four networks and compare them to predictions of Theorem 4.7.1. Table 4.6 shows the results. Note the predicted degree exponents remarkably agree with the true exponents, validating our model. This is interesting as we specified the model of temporal node behavior (lifetime+gaps) that results in an accurate structural network property (power law degree distribution).

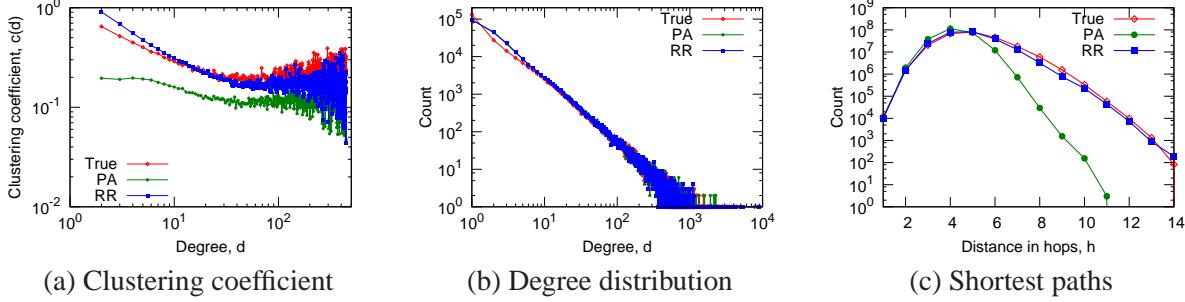


Figure 4.12: We take FLICKR network at first half of its evolution. Then we simulate the evolution using our model and PA for the second half, and compare the obtained networks with the real FLICKR network. Notice our model matches the macroscopic statistical properties of the true FLICKR network very well, and in fact much better than PA.

Figure 4.11 plots degree distributions of four networks and gives the power law fits. Table 4.6 shows the values of parameters λ , α and β measured from the evolution of the networks. We also show the measured degree exponent (denoted as true γ), and the degree exponent predicted from equation 4.1. Notice the remarkable agreement in degree exponent between the data and the model prediction.

For example, in FLICKR we observe the following parameters (see Table 4.6): $\lambda = 0.0092$, and $\alpha = 0.84$, $\beta = 0.0020$. Using equation 4.1 we obtain degree exponent $\gamma = 1.74$, which is very close to true exponent of 1.73 (see figure 4.11). See table 4.6 for comparison of true degree exponents and the degree exponents as predicted by our gap model.

We find this somewhat surprising as using only three parameters (1 parameter for node lifetime, and 2 for the gap distribution) we can accurately model the temporal part of the network evolution. Basically, with just 3 parameters we can accurately describe the non-structural part evolution (*i.e.*, everything except the selection of the edge destination).

4.7.3 Unfolding network evolution

To further our understanding of the network evolution, especially the edge creation process, we perform the following semi-simulation. We consider the real network $G_{T/2}$ and evolve it from $t = T/2, \dots, T$ using the random-random model to obtain a network G'_T . At the end of the evolution, we compare the macroscopic properties of G'_T and G_T . For completeness we also compare the results to the Preferential Attachment (PA) model.

More precisely, we evolve $G_{T/2}$ by considering all the edges that were created after time $T/2$ between the nodes in $G_{T/2}$. (We do not allow new nodes to join $G_{T/2}$.) We consider two different processes to place these new edges. In the first process (PA), we select two nodes preferentially, with probabilities proportional to their degrees, and add an edge. In the second process (RR), we use the random-random triangle-closing model, *i.e.*, we first select a node preferentially and then pick a node two hops away using the random-random model.

Figure 4.12 shows results for FLICKR: clustering coefficient, degree distribution, and pairwise distance histogram for the true data, and the two simulations. The random-random model matches the true network well and outperforms the PA. Similar results also hold for other networks; we omit these plots.

4.8 Discussion

In this chapter we presented a microscopic analysis of the edge-by-edge evolution of four large online social networks. The use of the maximum-likelihood principle allowed us to quantify the bias of new edges towards the degree and age of nodes, and to objectively compare various models such as preferential attachment. In fact, our work is the first to directly quantify the amount of preferential attachment that occurs in the evolution of large networks.

Our study shows that most new edges span very short distances, typically closing triangles. Motivated by these observations, we developed a complete model of network evolution, incorporating node arrivals, edge initiation, and edge destination selection processes. While node arrivals are mostly network-specific, the edge initiation process can be captured by exponential node lifetimes and a “gap” model based on a power law with exponential cutoff. We arrive at an extremely simple yet surprisingly accurate description of the edge destination selection in real networks. Moreover, our model is the first to accurately gives the complete picture of network evolution from node and edge arrivals to edge placement. Our model of network evolution can be used to generate arbitrary-sized synthetic networks that closely mimic the macroscopic characteristics of real social networks.

Chapter 5

Kronecker graphs

How can we generate realistic network? In addition, how can we do so with a mathematically tractable model that allows for rigorous analysis of network properties? Real networks exhibit a long list of surprising properties: Heavy tails for the in- and out-degree distribution; heavy tails for the eigenvalues and eigenvectors; small diameters; and over time the densification power law and shrinking diameters occur. The present network models and generators either fail to match several of the above properties, are complicated to analyze mathematically, or both. In this chapter we propose a generative model for networks that is both mathematically tractable and can generate networks that have all the above mentioned structural properties. Our main idea here is to use a non-standard matrix operation, the *Kronecker product*, to generate graphs that we refer to as “Kronecker graphs”.

First, we show that Kronecker graphs naturally obey common network properties; in fact, we rigorously *prove* that they do so. We also provide empirical evidence showing that Kronecker graphs can well mimic the structure of real networks.

Then, given a large real network, we present KRONFIT, a fast and scalable algorithm for fitting the Kronecker graph generation model to real networks. A naive approach to fitting would take super-exponential time. In contrast, KRONFIT takes *linear* time, by exploiting the structure of Kronecker matrix multiplication and by using statistical simulation techniques.

Experiments on large real and synthetic networks show that KRONFIT finds accurate parameters that indeed very well mimic the properties of target networks. Once fitted, the model parameters can be used to gain insights about the network structure, and the resulting synthetic graphs can be used for null-models, anonymization, extrapolations, and graph summarization.

5.1 Introduction

What do real graphs look like? How do they evolve over time? How can we generate synthetic, but realistic looking, time-evolving graphs? Recently network analysis has been attracting much interest, with an emphasis on finding patterns and abnormalities in social networks, computer networks, e-mail interactions, gene regulatory networks, and many more. Most of the work focuses on static snapshots of graphs, where fascinating “laws” have been discovered, including small diameters and heavy-tailed degree distributions.

As such structural “laws” have been discovered a natural next question is to find a model that produces networks with such structure. Thus, a good realistic network generation model is important for at least two reasons. The first is that it can generate graphs for extrapolations, “what-if” scenarios, and simulations, when real graphs are difficult or impossible to collect. For example, how well will a given protocol run on the Internet five years from now? Accurate network models can produce more realistic models for the future Internet, on which simulations can be run. The second reason is more subtle: it forces us to think about the network properties that a graph models should obey, to be realistic.

In this chapter we introduce Kronecker graphs, a network generative model which obeys all the main static network patterns that have appeared in the literature. Our model also obeys the temporal evolution patterns that we described in chapter 3. And, contrary to other models that match this combination of network properties, Kronecker graphs also lead to tractable analysis and rigorous proofs. Furthermore, Kronecker graphs generative process also has a nice natural interpretation and justification.

Our model is based on a matrix operation, the *Kronecker product*. There are several known theorems on Kronecker products, which correspond exactly to a significant portion of what we want to prove: heavy-tailed distributions for in-degree, out-degree, eigenvalues, and eigenvectors. We also demonstrate how a Kronecker Graph can match the behavior of several real networks (social networks, citations, web, internet, and others). While Kronecker products have been studied by the algebraic combinatorics community (see, e.g., [Chow, 1997]), the present work is the first to employ this operation in the design of network models to match real data.

Then we also make a step further and tackle the following problem: Given a large real network, we want to generate a synthetic graph, so that our resulting synthetic graph matches the properties of the real network as well as possible.

Ideally we would like: (a) A graph generation model that naturally produces networks with many properties that are also found in real networks. (b) The model parameter estimation should be fast and scalable, so that we can handle networks with millions of nodes. (c) The resulting set of parameters should generate realistic-looking networks that match the statistical properties of the target, real networks.

In general the problem of modeling network structure presents several conceptual and engineering challenges: Which generative model should we choose, among the many in the literature? How do we measure the goodness of the fit? (Least squares don’t work well for power laws, for subtle reasons!) If we use likelihood, (that we do), how to estimate it faster than in time quadratic on the number of nodes? How do we solve the node correspondence problem (which node of the real network corresponds to what node of the synthetic one)?

To answer the above questions we present KRONFIT, a fast and scalable algorithm for fitting Kronecker graphs by using the maximum likelihood principle. When calculating the likelihood there are two challenges: First, one needs to solve the node correspondence problem by matching the nodes of the real and the synthetic network. Essentially, one has to consider all mappings of nodes of the network to the rows and columns of the graph adjacency matrix. This becomes intractable for graphs with more than a handful of nodes. Even when given the “true” correspondences just evaluating the likelihood is still prohibitively expensive for the size of graphs we want to consider here. We present solutions to both of these problems: We develop Metropolis sampling algorithm for sampling node correspondences, and approximate the likelihood to obtain a *linear* time algorithm that scales to large networks with millions of nodes and edges. KRONFIT gives orders of magnitude speed-ups against older methods (20 minutes on a commodity PC, versus 2 days on a 50-machine cluster).

Our extensive experiments on synthetic and real networks show that Kronecker Graph can efficiently model statistical properties of networks, like degree distribution and diameter, while using only four parameters.

Once the model is fitted to the real network, there are several benefits and applications:

- (a) The parameters give us insight into the structure of the network itself;
- (b) *Extrapolations*: we can use the model to generate a larger graph, to help us understand how the network will look like in the future;
- (c) *Sampling*: conversely, we can also generate a smaller graph, which may be useful for running simulation experiments (*e.g.*, simulating routing algorithms in computer networks, or virus/worm propagation algorithms), when these algorithms may be too slow to run on large graphs;
- (d) *Null-model*: when working with network data we would often like to assess the significance or the extent to which a certain network property is expressed. We can use the fitted Kronecker graph as an accurate null-model.
- (e) *Simulations*: given an algorithm working on a graph we would like to evaluate how its performance depends on various properties of the network. Using our model one can generate graphs that exhibit various combinations of such properties, and then evaluate the algorithm.
- (f) *Graph compression*: we can compress the graph, by storing just the model parameters, and the deviations between the real and the synthetic graph;
- (g) *Anonymization*: suppose that the real graph cannot be publicized, like, *e.g.*, corporate e-mail network; customer-product sales in a recommendation system. Yet, we would like to share our network. Our work gives ways to such a realistic, 'similar' network.

The rest of the chapter is organized as follows: Section 5.2 briefly surveys the related literature. In section 5.3 we introduce the Kronecker graphs model, and give formal statements about the properties of networks it generates. We investigate the model using simulation in Section 5.4 and continue by introducing KRONFIT, the Kronecker graphs parameter estimation algorithm, in Section 5.5. We present experimental results on real and synthetic networks in Section 5.6. We close with discussion and conclusions in sections 5.7 and 5.8.

5.2 Relation to previous work on network modeling

Networks across a wide range of domains present surprising regularities, like power laws, small diameters, communities, and so on. We use these patterns as sanity checks, that is, our synthetic graphs should match those properties of the real target graph.

Most of the related work in this field has concentrated on two aspects: properties and patterns found in real-world networks, and then ways to find models to build understanding about the emergence of these properties. First, we will discuss the commonly found patterns in (static and temporally evolving) graphs, and finally, the state of the art in graph generation methods. Refer to chapter 2 for more detailed discussion of graph patterns and explanatory models.

5.2.1 Graph Patterns

Here we briefly introduce the network patterns (also referred to as properties or statistics) that we will later use to compare the similarity between the real networks and their synthetic counterparts produced by Kronecker graphs model. While many patterns have been discovered, two of the principal ones are heavy-tailed degree distributions and small diameters. Refer to chapter 2 for more details.

Degree distribution: The degree-distribution of a graph is a power law if the number of nodes N_d with degree d is given by $N_d \propto d^{-\gamma}$ ($\gamma > 0$) where γ is called the power law exponent. Power laws have been found in the Internet [Faloutsos et al., 1999], the Web [Kleinberg et al., 1999, Broder et al., 2000], citation graphs [Redner, 1998], online social networks [Chakrabarti et al., 2004] and many others.

Small diameter: Most real-world graphs exhibit relatively small diameter (the “small- world” phenomenon, or “six degrees of separation”): A graph has diameter D if every pair of nodes can be connected by a path of length at most D edges. The diameter D is susceptible to outliers. Thus, a more robust measure of the pair wise distances between nodes in a graph is the *effective diameter* [Tauro et al., 2001], which is the minimum number of links (steps/hops) in which some fraction (or quantile q , say $q = 0.9$) of all connected pairs of nodes can reach each other. The effective diameter has been found to be small for large real-world graphs, like Internet, Web, and online social networks [Albert and Barabási, 2002, Milgram, 1967, Leskovec et al., 2005b].

Hop-plot: extends the notion of diameter by plotting the number of reachable pairs $g(h)$ within h hops, as a function of the number of hops h [Palmer et al., 2002]. It gives us a sense of how quickly nodes’ neighborhoods expand with the number of hops.

Scree plot: This is a plot of the eigenvalues (or singular values) of the graph adjacency matrix, versus their rank, using the logarithmic scale. The scree plot is also often found to approximately obey a power law [Chakrabarti et al., 2004, Farkas et al., 2001]. Moreover, this pattern was also found analytically for random power law graphs [Chung et al., 2003a].

Network values: The distribution of eigenvector components (indicators of “network value”) associated to the largest eigenvalue of the graph adjacency matrix has also been found to be skewed [Chakrabarti et al., 2004].

Node triangle participation: is a measure of transitivity in networks. It counts the number of triangles a node participates in, *i.e.*, the number of connections between the neighbors of a node. The plot of the number of triangles Δ versus the number of nodes participating in Δ triangles has also been found to be skewed [Tsourakakis, 2008].

Densification Power Law: The relation between the number of edges $E(t)$ and the number of nodes $N(t)$ in evolving network at time t obeys the *densification power law* (DPL), which states that $E(t) \propto N(t)^a$. The *densification exponent* a is typically greater than 1, implying that the average degree of a node in the network is *increasing* over time. This means that real networks tend to sprout many more edges than nodes, and thus densify as they grow [Leskovec et al., 2005b, 2007b]. See chapter 3 for more details.

Shrinking diameter: The effective diameter of graphs tends to shrink or stabilize as the number of nodes in a network grows over time [Leskovec et al., 2005b, 2007b]. This is somewhat counterintuitive since from common experience as one would expect that as the volume of the object (a graph) grows, the size (*i.e.*, the diameter) would also grow. But for networks it seems this does not hold as the diameter shrinks and then stabilizes as the network grows. See chapter 3 for more details.

5.2.2 Generative models of network structure

The earliest probabilistic generative model for graphs was the Erdős-Rényi [Erdős and Rényi, 1960] random graph model, where each pair of nodes has an identical, independent probability of being joined by an edge. The study of this model has led to a rich mathematical theory; however, as the model was not developed to model real-world networks it produces graphs that fail to match real networks in a number of respects (for example, it does not produce heavy-tailed degree distributions).

The vast majority of recent network models involve some form of *preferential attachment* [Barabási and Albert, 1999, Albert and Barabási, 2002, Winick and Jamin, 2002, Kleinberg et al., 1999, Kumar et al., 1999a] that employs a simple rule: new node joins the graph at each time step, and then creates a connection to an existing node u with the probability proportional to the degree of the node u . This leads to the “rich get richer” phenomena and to power law tails in degree distribution. However, the diameter in this model grows slowly with the number of nodes N , which violates the “shrinking diameter” property mentioned above.

There are also many variations of preferential attachment model all somehow employing the “rich get richer” type mechanism. For example, “copying model” [Kumar et al., 2000], the “winner does not take all” model [Pennock et al., 2002], the “forest fire” model [Leskovec et al., 2005b], “random surfer model” [Blum et al., 2006], etc.

A different family of network methods strives for small diameter and local clustering in networks. Examples of such models include the *small-world* model [Watts and Strogatz, 1998] and the Waxman generator [Waxman, 1988]. Another family of models shows that heavy tails emerge if nodes try to optimize their connectivity under resource constraints [Carlson and Doyle, 1999, Fabrikant et al., 2002]. Refer to chapter 2 for further details on network models.

In summary, most current models focus on modeling only one (static) network property, and neglect the others. In addition, it is usually hard to analytically analyze properties of the network model. On the other hand, Kronecker graphs model we describe in the next section addresses these issues as it matches multiple properties of real networks at the same time, while being analytically tractable lending itself to rigorous analysis.

5.2.3 Parameter estimation of network models

Until recently relatively little effort was made to fit the above network models to real data. One of the difficulties is that most of the above models usually do not have a probabilistic interpretation, but rather define a mechanism or a principle by which a network is constructed.

Most work in estimating network models comes from the area of social sciences, statistics and social network analysis where the *exponential random graphs*, also known as $p*$ model, were introduced [Wasserman and Pattison, 1996]. The model essentially defines a log linear model over all possible graphs G , $p(G|\theta) \propto \exp(\theta^T s(G))$, where G is a graph, and s is a set of functions, that can be viewed as summary statistics for the structural features of the network. The $p*$ model usually focuses on “local” structural features of networks (like, e.g., characteristics of nodes that determine a presence of an edge, link reciprocity, etc.). As exponential random graphs have been very useful for modeling small networks, and individual nodes and edges our goal here is different in a sense that we aim to accurately model the structure of the network as a whole. Moreover, we aim to model and estimate parameters of networks with millions of

nodes, while even for graphs of small size (> 100 nodes) the number of model parameters in exponential random graphs usually becomes too large, and estimation prohibitively expensive, both in terms of computational time and memory.

Regardless of a particular choice of a network model, a common theme when estimating the likelihood $P(G)$ of a graph G under some model is the challenge of finding the correspondence between the nodes of the true network and its synthetic counterpart. The node correspondence problem results in the factorially many possible matchings of nodes. One can think of the correspondence problem as some kind of graph isomorphism test. Two isomorphic graphs G and G' with differently assigned node ids should have same likelihood $P(G) = P(G')$ so we aim to find an accurate mapping between the nodes of the two graphs.

Ordering or a permutation defines the mapping of nodes in one network to nodes in the other network. For example, Butts [Butts, 2005] used permutation sampling to determine similarity between two graph adjacency matrices, while Bezáková *et al.* [Bezáková et al., 2006] used permutations for graph model selection. Recently, an approach for estimating parameters of the “copying” model was introduced [Wiuf et al., 2006], however authors also note that the class of “copying” models may not be rich enough to accurately model real networks. As we show later, Kronecker graphs model seems to have the necessary expressive power to mimic real networks well.

5.3 Kronecker graphs model

The Kronecker graphs model we propose here is based on a recursive construction. Defining the recursion properly is somewhat subtle, as a number of standard, related graph construction methods fail to produce graphs that densify according to the patterns observed in real networks, and they also produce graphs whose diameters increase. To produce densifying graphs with constant/shrinking diameter, and thereby match the qualitative behavior of a real network, we develop a procedure that is best described in terms of the *Kronecker product* of matrices. To help in the description of the method, the accompanying table provides a list of symbols and their definitions.

5.3.1 Main idea

The main intuition behind the model is to create self-similar graphs, recursively. We begin with an *initiator* graph K_1 , with N_1 nodes and E_1 edges, and by recursion we produce successively larger graphs K_2, K_3, \dots such that the k^{th} graph K_k is on $N_k = N_1^k$ nodes. If we want these graphs to exhibit a version of the Densification Power Law [Leskovec et al., 2005b], then K_k should have $E_k = E_1^k$ edges. This is a property that requires some care in order to get right, as standard recursive constructions (for example, the traditional Cartesian product or the construction of [Barabási et al., 2001]) do not satisfy it.

It turns out that the *Kronecker product* of two matrices is the right tool for this goal. The Kronecker product is defined as follows:

SYMBOL	DESCRIPTION
G	Real network
N	Number of nodes in G
E	Number of edges in G
K	Kronecker graph (synthetic estimate of G)
K_1	Initiator of a Kronecker Graph
N_1	Number of nodes in initiator K_1
E_1	Number of edges in initiator K_1
$G \otimes H$	Kronecker product of adjacency matrices of graphs G and H
$K_1^{[k]} = K_k = K$	k^{th} Kronecker power of K_1
$K_1[i, j]$	Entry at row i and column j of K_1
$\Theta = \mathcal{P}_1$	Stochastic Kronecker initiator
$\mathcal{P}_1^{[k]} = \mathcal{P}_k = \mathcal{P}$	k^{th} Kronecker power of \mathcal{P}_1
$\theta_{ij} = \mathcal{P}_1[i, j]$	Entry at row i and column j of \mathcal{P}_1
$p_{ij} = \mathcal{P}_k[i, j]$	Probability of an edge (i, j) in \mathcal{P}_k , i.e., entry at row i and column j of \mathcal{P}_k
$K = R(\mathcal{P})$	Realization of a Stochastic Kronecker graph \mathcal{P}
$l(\Theta)$	Log-likelihood. Log-prob. that Θ generated real graph G , $\log P(G \Theta)$
$\hat{\Theta}$	Parameters at maximum likelihood, $\hat{\Theta} = \operatorname{argmax}_{\Theta} P(G \Theta)$
σ	Permutation that maps node ids of G to those of \mathcal{P}
a	Densification power law exponent, $E(t) \propto N(t)^a$
D	Diameter of a graph
N_c	Number of nodes in the largest weakly connected component of a graph
ω	Proportion of times SwapNodes permutation proposal distribution is used

Table 5.1: Table of symbols.

Definition 5.3.1 (Kronecker product of matrices). *Given two matrices $\mathbf{A} = [a_{i,j}]$ and \mathbf{B} of sizes $n \times m$ and $n' \times m'$ respectively, the Kronecker product matrix \mathbf{C} of dimensions $(n \cdot n') \times (m \cdot m')$ is given by*

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} \doteq \begin{pmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} & \dots & a_{1,m}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} & \dots & a_{2,m}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}\mathbf{B} & a_{n,2}\mathbf{B} & \dots & a_{n,m}\mathbf{B} \end{pmatrix} \quad (5.1)$$

We then define the Kronecker product of two graphs simply as the Kronecker product of their corresponding adjacency matrices.

Definition 5.3.2 (Kronecker product of graphs). *If G and H are graphs with adjacency matrices $A(G)$ and $A(H)$ respectively, then the Kronecker product $G \otimes H$ is defined as the graph with adjacency matrix $A(G) \otimes A(H)$.*

Observation 5.3.3 (Edges in Kronecker-multiplied graphs).

$$\text{Edge } (X_{ij}, X_{kl}) \in G \otimes H \text{ iff } (X_i, X_k) \in G \text{ and } (X_j, X_l) \in H$$

where X_{ij} and X_{kl} are nodes in $G \otimes H$, and X_i, X_j, X_k and X_l are the corresponding nodes in G and H , as in Figure 5.1.

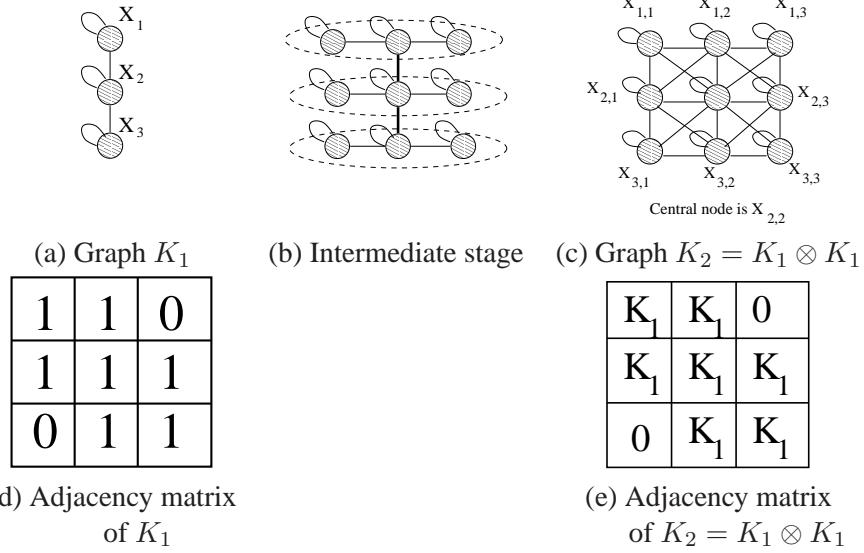


Figure 5.1: Example of Kronecker multiplication: Top: a “3-chain” initiator graph and its Kronecker product with itself; each of the X_i nodes gets expanded into 3 nodes, which are then linked using Observation 5.3.3. Bottom row: the corresponding adjacency matrices. See figure 5.2 for adjacency matrices of K_3 and K_4 .

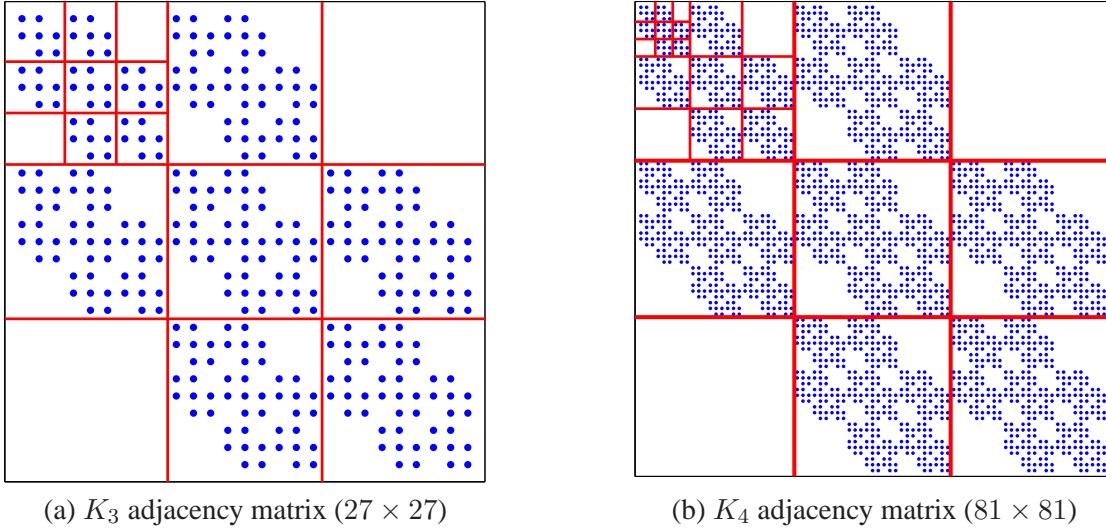


Figure 5.2: Adjacency matrices of K_3 and K_4 , the 3rd and 4th Kronecker power of K_1 matrix as defined in Figure 5.1. Dots represent non-zero matrix entries, and white space represents zeros. Notice the recursive self-similar structure of the adjacency matrix.

The last observation is subtle, but crucial, and deserves elaboration. Basically, each node in $G \otimes H$ can be represented as an ordered pair X_{ij} , with i a node of G and j a node of H , and with an edge joining X_{ij} and X_{kl} precisely when (X_i, X_k) is an edge of G and (X_j, X_l) is an edge of H . This is a direct consequence of the hierarchical nature of the Kronecker product. Figure 5.1(a–c) further illustrates this by showing the recursive construction of $G \otimes H$, when $G = H$ is a 3-node chain. Consider node $X_{1,2}$ in

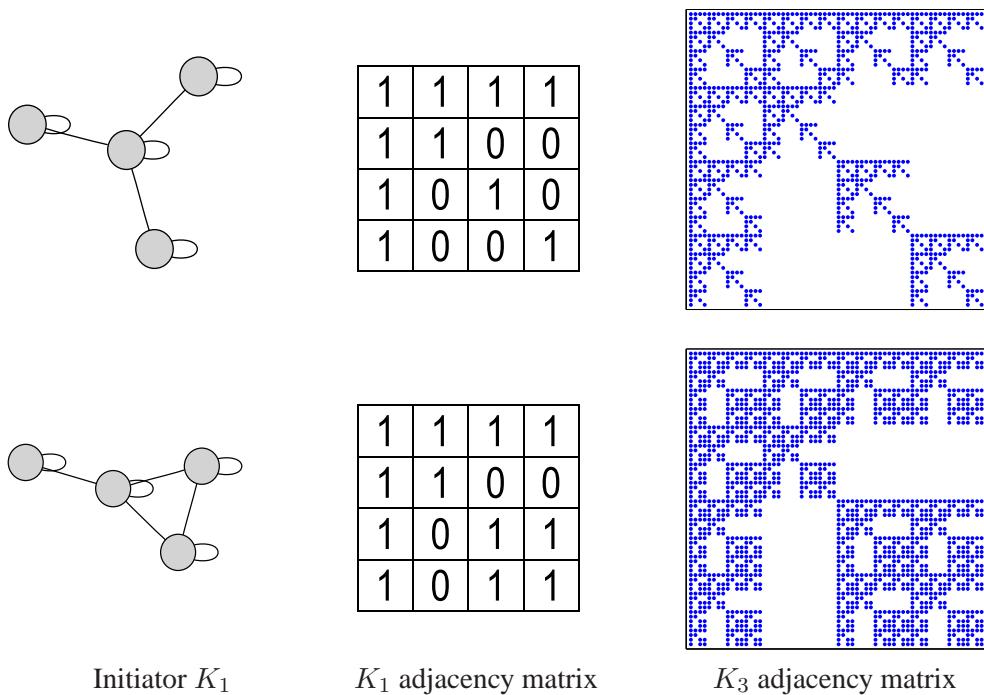


Figure 5.3: Two examples of Kronecker initiators on 4 nodes and the self-similar adjacency matrices they produce.

Figure 5.1(c): It belongs to the H graph that replaced node X_1 (see Figure 5.1(b)), and in fact is the X_2 node (*i.e.*, the center) within this small H -graph.

We propose to produce a growing sequence of matrices by iterating the Kronecker product:

Definition 5.3.4 (Kronecker power). *The k^{th} power of K_1 is defined as the matrix $K_1^{[k]}$ (abbreviated to K_k), such that:*

$$K_1^{[k]} = K_k = \underbrace{K_1 \otimes K_1 \otimes \dots \otimes K_1}_{k \text{ times}} = K_{k-1} \otimes K_1$$

Definition 5.3.5 (Kronecker graph). *Kronecker graph of order k is defined by the adjacency matrix $K_1^{[k]}$, where K_1 is the Kronecker initiator adjacency matrix.*

The self-similar nature of the Kronecker graph product is clear: To produce K_k from K_{k-1} , we “expand” (replace) each node of K_{k-1} by converting it into a copy of K_1 , and we join these copies together according to the adjacencies in K_{k-1} (see Figure 5.1). This process is very natural: one can imagine it as positing that communities within the graph grow recursively, with nodes in the community recursively getting expanded into miniature copies of the community. Nodes in the sub-community then link among themselves and also to nodes from other communities.

5.3.2 Analysis of Kronecker Graphs

We shall now discuss the properties of Kronecker graphs, specifically, their degree distributions, diameters, eigenvalues, eigenvectors, and time-evolution. Our ability to prove analytical results about all of these properties is a major advantage of Kronecker graphs over other network models.

Degree distribution

The next few theorems prove that several distributions of interest are *multinomial* for our Kronecker graph model. This is important, because a careful choice of the initial graph K_1 makes the resulting multinomial distribution to behave like a power law or DGX distribution [Bi et al., 2001, Clauset et al., 2007].

Theorem 5.3.6 (Multinomial degree distribution). *Kronecker graphs have multinomial degree distributions, for both in- and out-degrees.*

Proof. Let the initiator K_1 have the degree sequence d_1, d_2, \dots, d_{N_1} . Kronecker multiplication of a node with degree d expands it into N_1 nodes, with the corresponding degrees being $d \times d_1, d \times d_2, \dots, d \times d_{N_1}$. After Kronecker powering, the degree of each node in graph K_k is of the form $d_{i_1} \times d_{i_2} \times \dots \times d_{i_k}$, with $i_1, i_2, \dots, i_k \in (1 \dots N_1)$, and there is one node for each ordered combination. This gives us the multinomial distribution on the degrees of K_k . Note also that the degrees of nodes in K_k can be expressed as the k^{th} Kronecker power of the vector $(d_1, d_2, \dots, d_{N_1})$. \square

Spectral properties

Next we analyze the spectral properties of adjacency matrix of a Kronecker graph. We show that both the distribution of eigenvalues and distribution of component values of eigenvectors of graph adjacency matrix both follow multinomial distribution.

Theorem 5.3.7 (Multinomial eigenvalue distribution). *The Kronecker graph K_k has a multinomial distribution for its eigenvalues.*

Proof. Let K_1 have the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_{N_1}$. By properties of the Kronecker multiplication [Loan, 2000, Langville and Stewart, 2004], the eigenvalues of K_k are the k^{th} Kronecker power of the vector of eigenvalues of the initiator matrix, $(\lambda_1, \lambda_2, \dots, \lambda_{N_1})^{[k]}$. As in Theorem 5.3.6, the eigenvalue distribution is a multinomial. \square

A similar argument using properties of Kronecker matrix multiplication shows the following.

Theorem 5.3.8 (Multinomial eigenvector distribution). *The components of each eigenvector of the Kronecker graph K_k follow a multinomial distribution.*

Proof. Let K_1 have the eigenvectors $\vec{v}_1, \vec{v}_2, \dots, \vec{v}_{N_1}$. By properties of the Kronecker multiplication [Loan, 2000, Langville and Stewart, 2004], the eigenvectors of K_k are given by the k^{th} Kronecker power of the vector: $(\vec{v}_1, \vec{v}_2, \dots, \vec{v}_{N_1})$, which gives a multinomial distribution for the components of each eigenvector in K_k . \square

We have just covered several of the static graph patterns. Notice that the proofs were a direct consequences of the Kronecker multiplication properties.

Connectivity of Kronecker graphs

We now present a series of results on the connectivity of Kronecker graphs. We show, maybe a bit surprisingly, that even if a Kronecker initiator graph is connected its Kronecker power can in fact be disconnected.

Lemma 5.3.9. *If at least one of G and H is a disconnected graph, then $G \otimes H$ is also disconnected.*

Proof. Without loss of generality we can assume that G has two connected components, while H is connected. Figure 5.4(a) illustrates the corresponding adjacency matrix of G . Using the notation from observation 5.3.3 let graph let G have nodes X_1, \dots, X_n , where nodes $\{X_1, \dots, X_r\}$ and $\{X_{r+1}, \dots, X_n\}$ form the two connected components. Now, $G \otimes H$ has at least two connected components as there are no edges: $(X_{ij}, X_{kl}) \notin G \otimes H$ for $i \in \{1, \dots, r\}$, $k \in \{r+1, \dots, n\}$, and all j, l . This follows directly from observation 5.3.3 as (X_i, X_k) are not edges in G . \square

Actually it turns out that both G and H can be connected but $G \otimes H$ is still disconnected. The following theorem analyzes this case.

Theorem 5.3.10. *If both G and H are connected but bipartite, then $G \otimes H$ is disconnected, and each of the two connected components is again bipartite.*

Proof. Again without loss of generality let G be bipartite with two partitions $A = \{X_1, \dots, X_r\}$ and $B = \{X_{r+1}, \dots, X_n\}$, where edges exists only between the partitions, and no edges exist inside the partition: $(X_i, X_k) \notin G$ for $i, k \in A$ or $i, k \in B$. Similarly, let H also be bipartite with two partitions $C = \{X_1, \dots, X_s\}$ and $D = \{X_{s+1}, \dots, X_m\}$. Figures 5.4(b) and (c) illustrate the structure of the corresponding adjacency matrices.

Now, there will be two connected components in $G \otimes H$: 1st component will be composed of nodes $\{X_{ij}\} \in G \otimes H$, where $(i \in A, j \in D)$ or $(i \in B, j \in C)$. And similarly, 2nd component will be composed of nodes $\{X_{ij}\}$, where $(i \in A, j \in C)$ or $(i \in B, j \in D)$. Basically, there exist edges between node sets (A, D) and (B, C) , and similarly between (A, C) and (B, D) but not across the sets. To see this we have to analyze the cases using observation 5.3.3. For example, in $G \otimes H$ there exist edges between nodes (A, D) and (B, C) as there exist edges $(i, k) \in G$ for $i \in A, k \in B$, and $(j, l) \in H$ for $j \in C$ and $l \in D$. Similar is true for nodes (A, C) and (B, D) . However, there are no edges cross the two sets, e.g., nodes from (A, D) do not link to (A, C) , as there are no edges between nodes in A (since G is bipartite). See Figures 5.4(d) and 5.4(e) for a visual proof. \square

Note that bipartite graphs are triangle free and have no self-loops. For example, stars, chains, trees and cycles of even length are all examples of bipartite graphs. This means that one way to generate a connected Kronecker graphs is to require the initiator K_1 to be connected while not being bipartite. For example, initiator K_1 can have a self loops, or a triangles (a triple of connected nodes), which makes it non-bipartite, and ensures that K_k will be connected.

For the remainder of the chapter we will focus on the initiator graphs K_1 that have self loops on all of their nodes so that we ensure K_k to be connected.

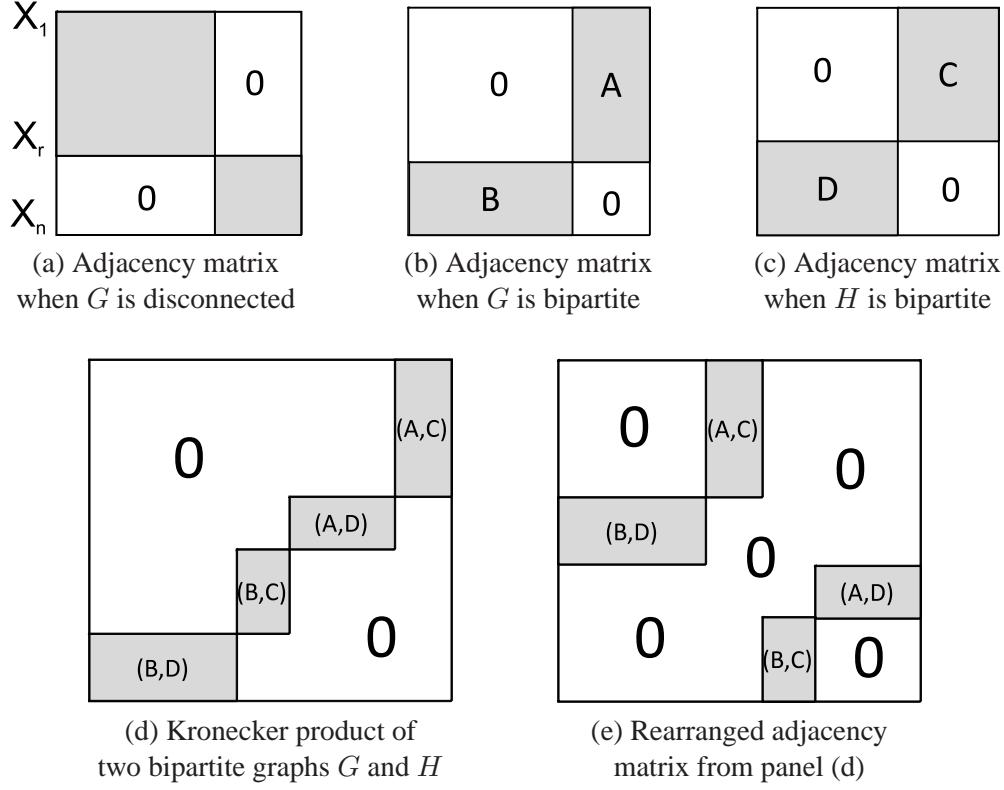


Figure 5.4: Graph adjacency matrices. Dark parts present connected (filled with ones) and white parts present empty (filled with zeros) parts of the adjacency matrix. (a) When G is disconnected, Kronecker multiplication with any matrix H will result in $G \otimes H$ being disconnected. (b) Adjacency matrix of a connected bipartite graph G with partitions A and B . (c) Adjacency matrix of a connected bipartite graph G with partitions C and D . (d) Kronecker product of two bipartite graphs G and H . (e) After rearranging the adjacency matrix $G \otimes H$ we clearly see the resulting graph is disconnected.

Temporal properties of Kronecker graphs

We continue with the analysis of temporal patterns of evolution of Kronecker graphs: the densification power law, and shrinking/stabilizing diameter [Leskovec et al., 2005b, 2007b].

Theorem 5.3.11 (Densification Power Law). *Kronecker graphs follow the Densification Power Law (DPL) with densification exponent $a = \log(E_1)/\log(N_1)$.*

Proof. Since the k^{th} Kronecker power K_k has $N_k = N_1^k$ nodes and $E_k = E_1^k$ edges, it satisfies $E_k = N_k^a$, where $a = \log(E_1)/\log(N_1)$. The crucial point is that this exponent a is independent of k , and hence the sequence of Kronecker powers follows an exact version of the Densification Power Law. \square

We now show how the Kronecker product also preserves the property of constant diameter, a crucial ingredient for matching the diameter properties of many real-world network datasets. In order to establish this, we will assume that the initiator graph K_1 has a self-loop on every node; otherwise, its Kronecker powers may be disconnected.

Lemma 5.3.12. *If G and H each have diameter at most D , and each has a self-loop on every node, then the Kronecker graph $G \otimes H$ also has diameter at most D .*

Proof. Each node in $G \otimes H$ can be represented as an ordered pair (v, w) , with v a node of G and w a node of H , and with an edge joining (v, w) and (x, y) precisely when (v, x) is an edge of G and (w, y) is an edge of H . (Note this exactly the Observation 5.3.3.) Now, for an arbitrary pair of nodes (v, w) and (v', w') , we must show that there is a path of length at most D connecting them. Since G has diameter at most D , there is a path $v = v_1, v_2, \dots, v_r = v'$, where $r \leq D$. If $r < D$, we can convert this into a path $v = v_1, v_2, \dots, v_D = v'$ of length exactly D , by simply repeating v' at the end for $D - r$ times. By an analogous argument, we have a path $w = w_1, w_2, \dots, w_D = w'$. Now by the definition of the Kronecker product, there is an edge joining (v_i, w_i) and (v_{i+1}, w_{i+1}) for all $1 \leq i \leq D - 1$, and so $(v, w) = (v_1, w_1), (v_2, w_2), \dots, (v_D, w_D) = (v', w')$ is a path of length D connecting (v, w) to (v', w') , as required. \square

Theorem 5.3.13. *If K_1 has diameter D and a self-loop on every node, then for every k , the graph K_k also has diameter D .*

Proof. This follows directly from the previous lemma, combined with induction on k . \square

As defined in section 2.1.2 we also consider the *effective diameter* D^* ; we defined the q -effective diameter as the minimum D^* such that, for at least a q fraction of the reachable node pairs, the path length is at most D^* . The q -effective diameter is a more robust quantity than the diameter, the latter being prone to the effects of degenerate structures in the graph (e.g., very long chains); however, the q -effective diameter and diameter tend to exhibit qualitatively similar behavior. For reporting results in subsequent sections, we will generally consider the q -effective diameter with $q = 0.9$, and refer to this simply as the *effective diameter*.

Theorem 5.3.14 (Effective Diameter). *If K_1 has diameter D and a self-loop on every node, then for every q , the q -effective diameter of K_k converges to D (from above) as k increases.*

Proof. To prove this, it is sufficient to show that for two randomly selected nodes of K_k , the probability that their distance is D converges to 1 as k goes to infinity.

We establish this as follows. Each node in K_k can be represented as an ordered sequence of k nodes from K_1 , and we can view the random selection of a node in K_k as a sequence of k independent random node selections from K_1 . Suppose that $v = (v_1, \dots, v_k)$ and $w = (w_1, \dots, w_k)$ are two such randomly selected nodes from K_k . Now, if x and y are two nodes in K_1 at distance D (such a pair (x, y) exists since K_1 has diameter D), then with probability $1 - (1 - 2/N_1)^k$, there is some index j for which $\{v_j, w_j\} = \{x, y\}$. If there is such an index, then the distance between v and w is D . As the expression $1 - (1 - 2/N_1)^k$ converges to 1 as k increases, it follows that the q -effective diameter is converging to D . \square

5.3.3 Stochastic Kronecker Graphs

While the Kronecker power construction discussed so far yields graphs with a range of desired properties, its discrete nature produces “staircase effects” in the degrees and spectral quantities, simply because individual values have large multiplicities. For example, degree distribution and distribution of eigenvalues of graph adjacency matrix and the distribution of the principal eigenvector components (*i.e.*, the “network”

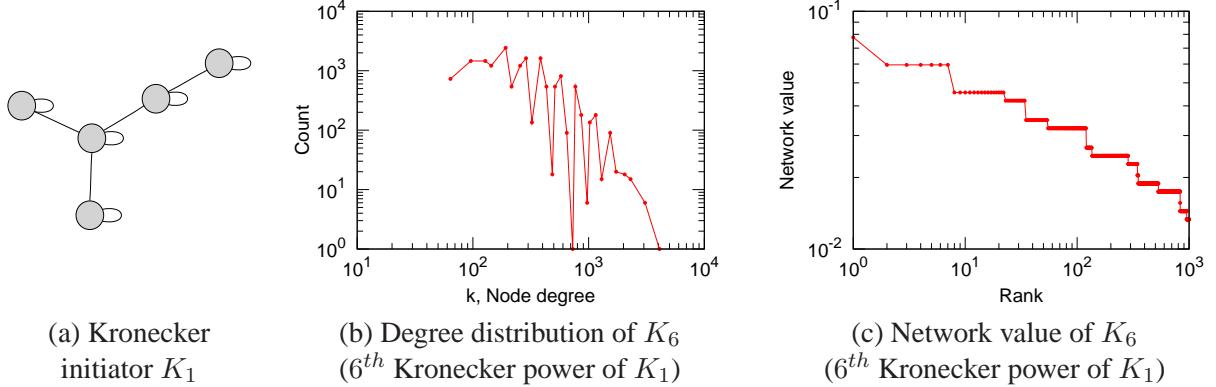


Figure 5.5: The “staircase” effect. Kronecker initiator and the degree distribution and network value plot for the 6th Kronecker power of the initiator. Notice the non-smoothness of the curves.

value) are all impacted by this. These quantities are multinomially distributed which leads to individual values with large multiplicities. Figure 5.5 illustrates the staircase effect.

Here we propose a stochastic version of Kronecker graphs that eliminates this effect. There are many possible ways how one could introduce stochasticity into Kronecker graphs model. Before introducing the proposed model, we introduce two simple ways of introducing randomness to Kronecker graphs and describe why they do not work.

Probably the simplest (but wrong) idea is to generate a large deterministic Kronecker graph K_k , and then uniformly at random flip some edges, *i.e.*, uniformly at random select entries of the graph adjacency matrix and flip them ($1 \rightarrow 0, 0 \rightarrow 1$). However, this will not work, as it will essentially superimpose a Erdős-Rényi random graph, which would, for example, corrupt the degree distribution – real networks usually have heavy tailed degree distributions, while random graphs have Binomial degree distributions. Second idea could be to allow weighted initiator matrix, *i.e.*, values of entries of K_1 are not restricted to values $\{0, 1\}$ but rather can be any non-negative real number. Using such K_1 one would generate K_k and then threshold the K_k matrix to obtain a binary adjacency matrix K , *i.e.*, for a chosen value of ϵ set $K[i, j] = 1$ if $K_k[i, j] > \epsilon$ else $K[i, j] = 0$. This also would not work as the mechanism would selectively remove edges and thus the low degree nodes which would have low weight edges would get isolated first.

Now we define *Stochastic Kronecker Graphs* model that overcomes the above issues. A more natural way to introduce stochasticity to Kronecker graphs is to relax the assumption that entries of the initiator matrix take only binary values. Now, we will allow cells of the initiator to take values on interval $[0, 1]$. This means now each entry of the initiator matrix encodes the probability of that particular edge appearing. We then Kronecker power such initiator matrix to obtain a large stochastic adjacency matrix, where again each entry of the large matrix gives the probability of that particular edge appearing in a big graph. Such stochastic adjacency matrix effectively defines a probability distribution over all graphs. To obtain a graph we simply sample an instance from this distribution by sampling individual edges, where each edge appears independently with probability given by the entry of the large stochastic adjacency matrix. More formally, we define:

Definition 5.3.15 (Stochastic Kronecker Graph). *Let \mathcal{P}_1 be a $N_1 \times N_1$ probability matrix: the value $\theta_{ij} \in \mathcal{P}_1$ denotes the probability that edge (i, j) is present, $\theta_{ij} \in [0, 1]$.*

Then k^{th} Kronecker power $\mathcal{P}_1^{[k]} = \mathcal{P}_k$, where each entry $p_{uv} \in \mathcal{P}_k$ encodes the probability of an edge (u, v) .

To obtain a graph, an instance (or realization), $K = R(\mathcal{P}_k)$ we include edge (u, v) in K with probability p_{uv} , $p_{uv} \in \mathcal{P}_k$.

First, note that sum of the entries of \mathcal{P}_1 , $\sum_{ij} \theta_{ij}$, can be greater than 1. Second, notice that in principle it takes $O(N_1^{2k})$ time to generate an instance K of a Stochastic Kronecker graph from the probability matrix \mathcal{P}_k . This means the time to get a realization K is quadratic in the size of \mathcal{P}_k as one has to flip a coin for each possible edge in the graph. Later we show how to generate Stochastic Kronecker graphs much faster, in the time *linear* in the number of edges in \mathcal{P}_k .

Probability of an edge

For the size of the graphs we aim to model and generate here taking \mathcal{P}_1 (or K_1) and then explicitly performing the Kronecker product of the initiator matrix is infeasible. The reason for this is that \mathcal{P}_1 is usually dense, so \mathcal{P}_k is also dense and one can not store it in memory. However, due to the structure of Kronecker multiplication one can easily computer the probability of an edge in \mathcal{P}_k .

The probability p_{uv} of an edge (u, v) occurring in k -th Kronecker power $\mathcal{P} = \mathcal{P}_k$ can be calculated in $O(k)$ time as follows:

$$p_{uv} = \prod_{i=0}^{k-1} \mathcal{P} \left[\left\lfloor \frac{u-1}{N_1^i} \right\rfloor (\bmod N_1) + 1, \left\lfloor \frac{v-1}{N_1^i} \right\rfloor (\bmod N_1) + 1 \right] \quad (5.2)$$

The equation imitates recursive descent into the matrix \mathcal{P} , where at every level i the appropriate entry of \mathcal{P}_1 is chosen. Since \mathcal{P} has N_1^k rows and columns it takes $O(k \log N_1)$ to evaluate the equation. Refer to figure 5.6 for the illustration of the recursive structure of \mathcal{P} .

5.3.4 Additional properties of Kronecker graphs

Stochastic Kronecker Graphs with initiator matrix of size $N_1 = 2$ were studied by Mahdian and Xu [Mahdian and Xu, 2007]. Authors show a phase transition for the emergence of the giant component and another phase transition for connectivity, and prove that such graphs have constant diameters beyond the connectivity threshold, but are not searchable using a decentralized algorithm [Kleinberg, 1999b].

Moreover, recently [Tsourakakis, 2008] gave a closed form expression for the number of triangles in a Kronecker graph that depends on the eigenvalues of the initiator graph K_1 .

5.3.5 Two interpretations of Kronecker graphs

Next, we present two natural interpretations of the generative process behind the Kronecker Graphs that go beyond the purely mathematical construction of Kronecker Graphs as introduced so far.

We already mentioned the first interpretation when we first defined Kronecker Graphs. One intuition is that networks and communities in them grow recursively, creating miniature copies of themselves.

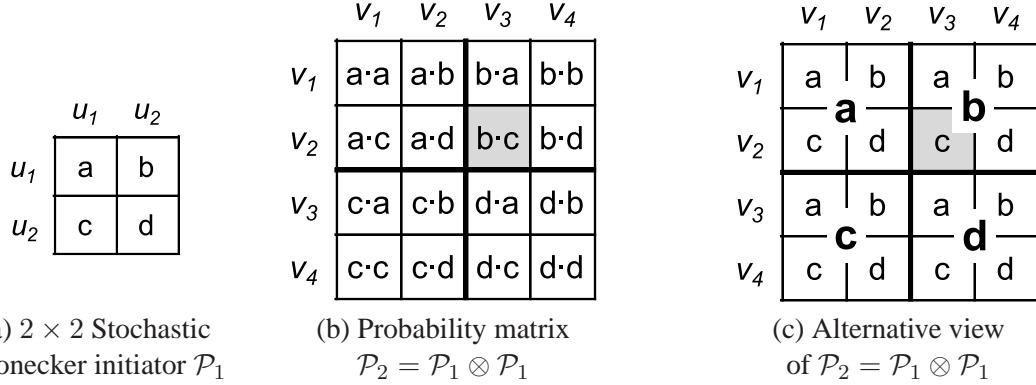


Figure 5.6: Stochastic Kronecker initiator \mathcal{P}_1 and the corresponding 2^{nd} Kronecker power \mathcal{P}_2 . Notice the recursive nature of the Kronecker product, with edge probabilities in \mathcal{P}_2 simply being products of entries of \mathcal{P}_1 .

Figure 5.1 depicts the process of the recursive community expansion. In fact, several researchers have argued that real networks are hierarchically organized [Ravasz et al., 2002, Ravasz and Barabási, 2003] and algorithms to extract the network hierarchical structure have also been developed [Sales-Pardo et al., 2007, Clauset et al., 2008]. Moreover, especially web graphs [Dill et al., 2002, Dorogovtsev et al., 2002, Crovella and Bestavros, 1997] and biological networks [Ravasz and Barabási, 2003] were found to be self-similar and “fractal”.

Second intuition comes from viewing every node of \mathcal{P}_k as being described with an ordered sequence of k nodes from \mathcal{P}_1 . (This is similar to the Observation 5.3.3 and the proof of Theorem 5.3.14.)

Let’s label nodes of the initiator matrix $\mathcal{P}_1, u_1, \dots, u_{N_1}$, and nodes of \mathcal{P}_k as $v_1, \dots, v_{N_1^k}$. Then every node v_i of \mathcal{P}_k is described with a sequence $(v_i(1), \dots, v_i(k))$ of node labels of \mathcal{P}_1 , where $v_i(l) \in \{u_1, \dots, u_k\}$. Similarly, consider also a second node v_j with the label sequence $(v_j(1), \dots, v_j(k))$. Then the probability p_e of an edge (v_i, v_j) in \mathcal{P}_k is exactly:

$$p_e(v_i, v_j) = \mathcal{P}_k[v_i, v_j] = \prod_{l=1}^k \mathcal{P}_1[v_i(l), v_j(l)]$$

(Note this is exactly the Equation 5.2.)

Now one can look at the description sequence of node v_i as a k dimensional vector of attribute values $(v_i(1), \dots, v_i(k))$. Then $p_e(v_i, v_j)$ is exactly the coordinate-wise product of appropriate entries of \mathcal{P}_1 , where the node description sequence selects which entries to multiply. Thus, the \mathcal{P}_1 matrix can be thought of as the attribute similarity matrix, *i.e.*, it encodes the probability of linking given that two nodes agree/disagree on the attribute value. Then the probability of an edge is simply a product of individual attribute similarities over the $k N_1$ -ary attributes that describe each of the two nodes.

This gives us a very natural interpretation of Stochastic Kronecker graphs: Each node is described by a sequence of categorical attribute values or features. And then the probability of two nodes linking depends on the product of individual attribute similarities. This way Kronecker graphs can effectively model homophily (nodes with similar attribute values are more likely to link) by \mathcal{P}_1 having high value entries on the diagonal; or heterophily (nodes that differ are more likely to link) by \mathcal{P}_1 having high entries off the diagonal.

Figure 5.6 shows an example. Let's label nodes of \mathcal{P}_1 u_1, u_2 as in Figure 5.6(a). Then every node of \mathcal{P}_k is described with an ordered sequence of k binary attributes. For example, Figure 5.6(b) shows an instance for $k = 2$ where node v_2 of \mathcal{P}_2 is described by (u_1, u_2) , and similarly v_3 by (u_2, u_1) . Then as shown in Figure 5.6(b), the probability of edge $p_e(v_2, v_3) = b \cdot c$, which is exactly $\mathcal{P}_1[u_2, u_1] \cdot \mathcal{P}_1[u_1, u_2] = b \cdot c$ — the product of entries of \mathcal{P}_1 , where the corresponding elements of the description of nodes v_2 and v_3 act as selectors of which entries of \mathcal{P}_1 to multiply.

Figure 5.6(c) further illustrates the recursive nature of Kronecker graphs. One can see Kronecker product as recursive descent into the big adjacency matrix where at each stage one of the entries or blocks is chosen. For example, to get to entry (v_2, v_3) one first needs to dive into quadrant b following by the quadrant c . This intuition will help us in section 5.3.6 to devise a fast algorithm for generating Kronecker graphs.

However, there are also two notes to make here. First, using a single initiator \mathcal{P}_1 we are implicitly assuming that there is one single and universal attribute similarity matrix that holds across all $k N_1$ -ary attributes. One can easily relax this assumption by taking a different initiator matrix for each attribute (initiator matrices can even be of different sizes as attributes are of different arity), and then Kronecker multiplying them to obtain a large network. Here each initiator matrix plays the role of attribute similarity matrix for that particular attribute.

For simplicity and convenience we will work with a single initiator matrix but all our methods can be trivially extended to handle multiple initiator matrices. Moreover, as we will see later in section 5.6 even a single 2×2 initiator matrix seems to be enough to capture large scale statistical properties of real-world networks.

Second assumption is harder to relax. When describing every node v_i with a sequence of attribute values we are implicitly assuming the values of all attributes are uniformly distributed (have same proportions), and that every node has a unique combination of attribute values. So, all possible combinations of attribute values are taken. For example, node v_1 in a large \mathcal{P}_k has attribute sequence (u_1, u_1, \dots, u_1) , v_{N_1} has $(u_1, u_1, \dots, u_1, u_{N_1})$, while the “last” node $v_{N_1^k}$ has attribute values $(u_{N_1}, u_{N_1}, \dots, u_{N_1})$. One can think of this as counting in N_1 -ary number system, where node attribute descriptions range from 0 (*i.e.*, “leftmost” node with attribute description (u_1, u_1, \dots, u_1)) to N_1^k (*i.e.*, “rightmost” node attribute description $(u_{N_1}, u_{N_1}, \dots, u_{N_1})$).

A simple way to relax the above assumption is to take a larger initiator matrix with a smaller number of parameters than the number of entries. This means that multiple entries of \mathcal{P}_1 will share the same value (parameter). For example, if attribute u_1 takes one value 66% of the times, and the other value 33% of the times, then one can model this by taking a 3×3 initiator matrix with only four parameters. Adopting the naming convention of Figure 5.6 this means that parameter a now occupies a 2×2 block, which then also makes b and c occupy 2×1 and 1×2 blocks, and d a single cell. This way one gets a four parameter model with uneven feature value distribution.

We note that the view of Kronecker graphs where every node is described with a set of features and the initiator matrix encodes the probability of linking given the attribute values of two nodes somewhat resembles the Random dot product graphs model [Young and Scheinerman, 2007, Nickel, 2008]. The important difference here is that we multiply individual linking probabilities, while in Random dot product graphs one takes the sum of individual probabilities which seems somewhat less natural.

5.3.6 Fast generation of Stochastic Kronecker Graphs

The intuition for fast generation of Stochastic Kronecker Graphs comes from the recursive nature of the Kronecker product and is closely related to the R-Mat graph generator [Chakrabarti et al., 2004]. Generating a Stochastic Kronecker graph K on N nodes naively takes $O(N^2)$ time. Here we present a linear time $O(E)$ algorithm, where E is the (expected) number of edges in K .

Figure 5.6(c) shows the recursive nature of the Kronecker product. To “arrive” to a particular edge (v_i, v_j) of \mathcal{P}_k one has to make a sequence of k (in our case $k = 2$) decisions among the entries of \mathcal{P}_1 , multiply the chosen entries of \mathcal{P}_1 , and then placing the edge (v_i, v_j) with the obtained probability.

Instead of flipping $O(N^2) = O(N_1^{2k})$ biased coins to determine the edges, we can place E edges by directly simulating the recursion of the Kronecker product. Basically we recursively choose sub-regions of matrix K with probability proportional to θ_{ij} , $\theta_{ij} \in \mathcal{P}_1$ until in k steps we descend to a single cell of the matrix and place an edge. For example, for (v_2, v_3) in Figure 5.6(c) we first have to choose b following by c .

As probability of each individual edge of \mathcal{P}_k follows a Bernoulli distribution, as the edge occurrences are independent, the number of edges in \mathcal{P}_k is Binomially distributed with mean $(\sum \theta_{ij})^k = E_1^k$, where $\theta_{ij} \in \mathcal{P}_1$. So, given a stochastic initiator matrix \mathcal{P}_1 we first sample the expected number of edges E in \mathcal{P}_k from a Binomial distribution. Then we place E edges in a graph K , by applying the recursive descent for k steps where at each step we choose entry (i, j) with probability θ_{ij}/E_1 where $\theta_{ij} \in \mathcal{P}_1$ and $E_1 = \sum_{ij} \theta_{ij}$. Since we add $E = E_1^k$ edges, the probability that edge (v_i, v_j) appears in K is exactly $\mathcal{P}_k[v_i, v_j]$. This basically means that in Stochastic Kronecker Graphs the initiator matrix encodes both the total number of edges in a graph and their structure. $\sum \theta_{ij}$ encodes the number of edges in the graph, while the proportions (ratios) of values θ_{ij} define how many edges each part of graph adjacency matrix will contain.

In practice it can happen that more than one edge lands in the same (v_i, v_j) cell of K . Even though values of \mathcal{P}_1 are usually skewed, adjacency matrices of real network are sparse which mitigates the problem.

5.3.7 Observations and connections

Next, we describe several observations about the properties of Kronecker graphs and make connections to other network models.

- *Bipartite graphs:* Kronecker Graphs can naturally model bipartite graphs. Instead of starting with a square $N_1 \times N_1$ initiator matrix, one can choose arbitrary $N_1 \times M_1$ initiator matrix, where rows define “left”, and columns the “right” side of the bipartite graph. Kronecker multiplication will then generate bipartite graphs with partition sizes N_1^k and M_1^k .
- *Graph distributions:* \mathcal{P}_k defines a distribution over all graphs, as it encodes the probability of all possible N_1^{2k} edges appearing in a graph by using exponentially smaller number of parameters (just N_1^2). As we will later see even a very small number of parameters, e.g., 4 (2×2 initiator matrix) or 9 (3×3 initiator), is enough to accurately model the structure large networks.
- *Natural extension of Erdős-Rényi random graph model:* Stochastic Kronecker Graphs represent a natural extension of Erdős-Rényi [Erdős and Rényi, 1960] random graphs. If one takes $\mathcal{P}_1 = [\theta_{ij}]$,

where every $\theta_{ij} = p$ then we obtain exactly the Erdős-Rényi model of random graphs $G_{n,p}$, where every node appears independently with probability p .

- *Relation to R-mat model:* The recursive nature of Stochastic Kronecker Graphs makes them related to the R-mat generator [Chakrabarti et al., 2004]. The difference between the two models is that in R-mat one needs to separately specify the number of edges, while in Stochastic Kronecker Graphs initiator matrix \mathcal{P}_1 also encodes the number of edges in the graph. Section 5.3.6 built on this similarity to devise a fast algorithm for generating Stochastic Kronecker graphs.
- *Densification:* Similarly as with deterministic Kronecker graphs the number of nodes in a Stochastic Kronecker Graph grows as N_1^k , and the expected number of edges grows as $(\sum_{ij} \theta_{ij})^k$. This means one would want to choose values θ_{ij} of the initiator matrix \mathcal{P}_1 so that $\sum_{ij} \theta_{ij} > N_1$ in order for the resulting network to densify.

5.4 Simulations of Kronecker graphs

In previous section we proved and now we demonstrate using simulation the ability of Kronecker graphs to match the patterns of real-world networks. We will tackle the problem of estimating the Kronecker Graphs model from real data, *i.e.*, finding the most likely initiator \mathcal{P}_1 , in the next section. Instead here we present simulation experiments using Kronecker graphs to explore the parameter space, and to compare properties of Kronecker Graphs to those found in large real networks.

5.4.1 Comparison to real graphs

We observe two kinds of graph patterns — “static” and “temporal.” As mentioned earlier, common static patterns include degree distribution, scree plot (eigenvalues of graph adjacency matrix vs. rank) and distribution of components of the principal eigenvector of graph adjacency matrix. Temporal patterns include the diameter over time, and the densification power law. For the diameter computation, we use the effective diameter as defined in Section 2.1.2.

For the purpose of this section consider the following setting. Given a real graph G we want to find Kronecker initiator that produces qualitatively similar graph. In principle one could try choosing each of the N_1^2 parameters for the matrix \mathcal{P}_1 separately. However, we reduce the number of parameters from N_1^2 to just two: α and β . Let K_1 be the initiator matrix (binary, deterministic); we create the corresponding stochastic initiator matrix \mathcal{P}_1 by replacing each “1” and “0” of K_1 with α and β respectively ($\beta \leq \alpha$). The resulting probability matrices maintain — with some random noise — the self-similar structure of the Kronecker graphs in the previous section (which, for clarity, we call *deterministic Kronecker graphs*). We defer the discussion of how to estimate \mathcal{P}_1 from data G to the next section.

The datasets we use here are:

- CIT-HEP-TH: This is a citation graph for High-Energy Physics Theory research papers from pre-print archive ArXiv, with a total of $N = 29,555$ papers and $E = 352,807$ citations [Gehrke et al., 2003]. We follow its evolution from January 1993 to April 2003, with one data-point per month.
- AS-ROUTEVIEWS: We also analyze a static dataset consisting of a single snapshot of connectivity among Internet Autonomous Systems [RouteViews, 1997] from January 2000, with $N = 6,474$

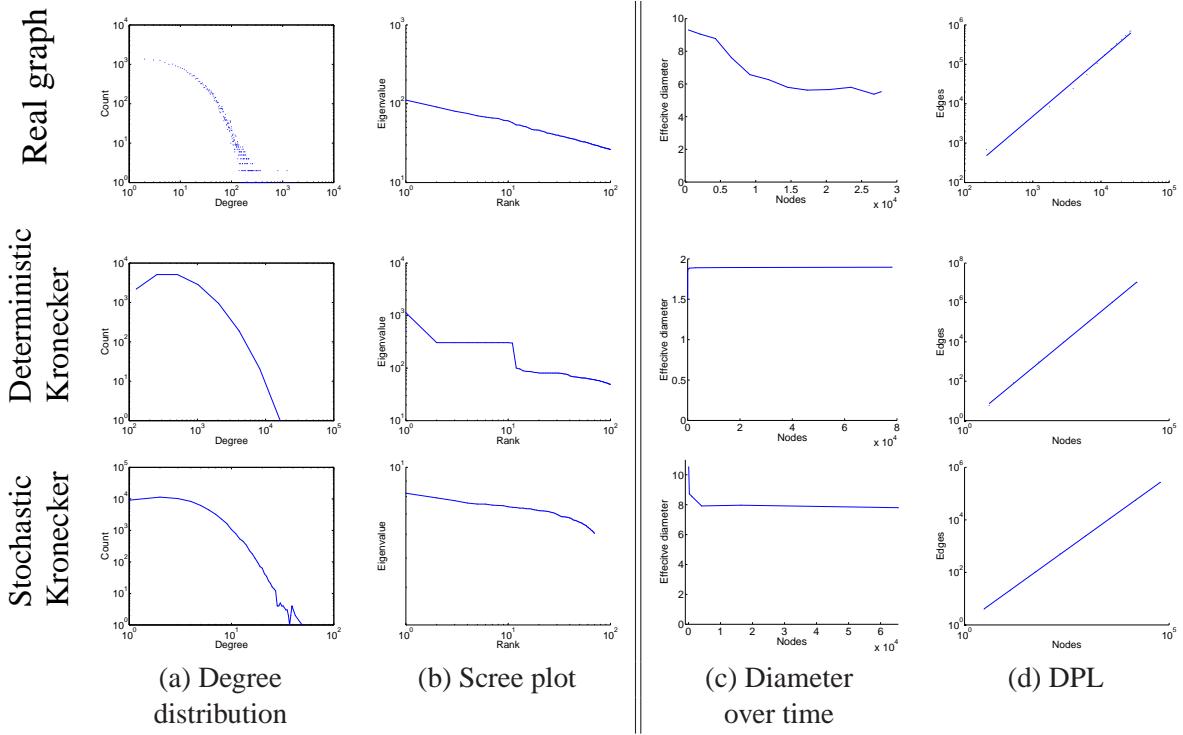


Figure 5.7: Citation network (CIT-HEP-TH): Patterns from the real graph (top row), the deterministic Kronecker graph with K_1 being a star graph on 4 nodes (center + 3 satellites) (middle row), and the Stochastic Kronecker graph ($\alpha = 0.41$, $\beta = 0.11$ – bottom row). *Static* patterns: (a) is the PDF of degrees in the graph (log-log scale), and (b) the distribution of eigenvalues (log-log scale). *Temporal* patterns: (c) gives the effective diameter over time (linear-linear scale), and (d) is the number of edges versus number of nodes over time (log-log scale). Notice that the Stochastic Kronecker Graph qualitatively matches all the patterns very well.

and $E = 26,467$.

Results are shown in Figure 5.7 for the CIT-HEP-TH graph which evolves over time. We show the plots of two static and two temporal patterns. We see that the deterministic Kronecker model already captures the qualitative structure of the degree and eigenvalue distributions, as well as the temporal patterns represented by the Densification Power Law and the stabilizing diameter. However, the deterministic nature of this model results in “staircase” behavior, as shown in scree plot for the deterministic Kronecker graph of Figure 5.7 (column (b), second row). We see that the Stochastic Kronecker Graphs smooth out these distributions, further matching the qualitative structure of the real data; they also match the shrinking-before-stabilization trend of the diameters of real graphs.

Similarly, Figure 5.8 shows plots for the static patterns in the *Autonomous systems* (AS-ROUTEVIEWS) graph. Recall that we analyze a single, static network snapshot in this case. In addition to the degree distribution and scree plot, we also show two typical plots [Chakrabarti et al., 2004]: the distribution of *network values* (principal eigenvector components, sorted, versus rank) and the *hop-plot* (the number of reachable pairs $g(h)$ within h hops or less, as a function of the number of hops h). Notice that, again, the Stochastic Kronecker graph matches well the properties of the real graph.

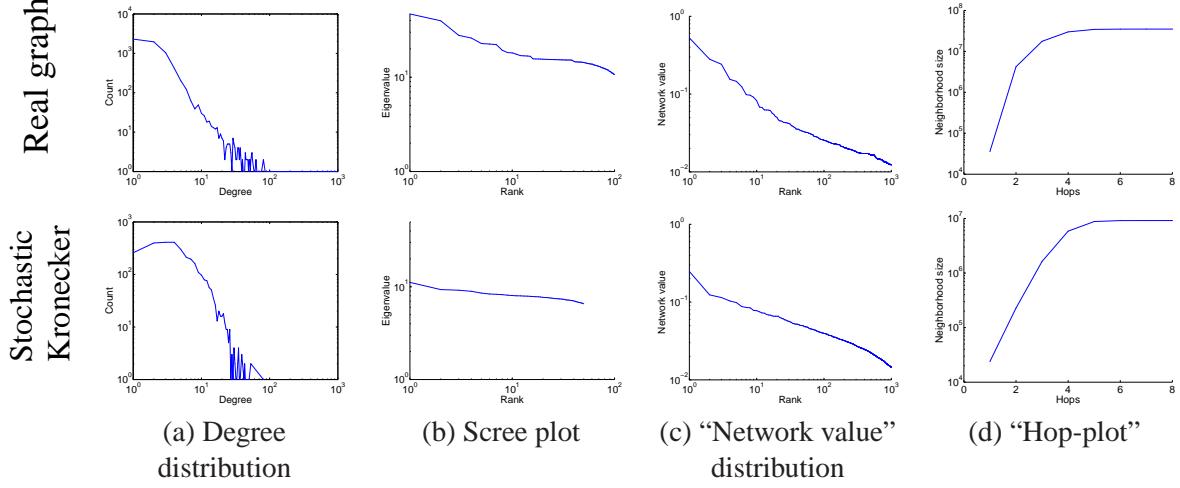


Figure 5.8: Autonomous systems (As-ROUTEVIEWS): Real (top) versus Kronecker (bottom). Columns (a) and (b) show the degree distribution and the scree plot, as before. Columns (c) and (d) show two more static patterns (see text). Notice that, again, the Stochastic Kronecker graph matches well the properties of the real graph.

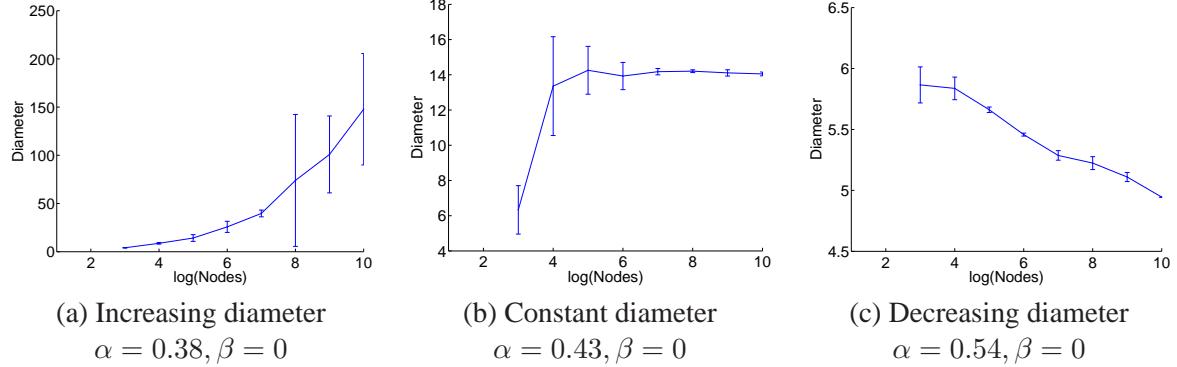


Figure 5.9: Diameter over time for a 4-node chain initiator graph. After each consecutive Kronecker power we measure the effective diameter. We use different settings of α parameter. $\alpha = 0.38, 0.43, 0.54$ and $\beta = 0$, respectively.

5.4.2 Parameter space of Kronecker Graphs

Last we present simulation experiments that investigate the parameter space of Stochastic Kronecker Graphs.

First, in Figure 5.9 we show the ability of Kronecker Graphs to generate networks with increasing, constant and decreasing/stabilizing effective diameter. We start with a 4-node chain initiator graph, setting each “1” of K_1 to α and each “0” to $\beta = 0$ to obtain \mathcal{P}_1 that we then use to generate a growing sequence of graphs. We plot the effective diameter of each $R(\mathcal{P}_k)$ as we generate a sequence of growing graphs $R(\mathcal{P}_2), R(\mathcal{P}_3), \dots, R(\mathcal{P}_{10})$. $R(\mathcal{P}_{10})$ has exactly 1,048,576 nodes. Notice Stochastic Kronecker graphs is a very flexible model. When the generated graph is very sparse (low value of α) we obtain graphs with slowly increasing effective diameter (Figure 5.9(a)). For intermediate values of α we get graphs with constant diameter (Figure 5.9(b)) and that in our case also slowly densify with densification exponent in

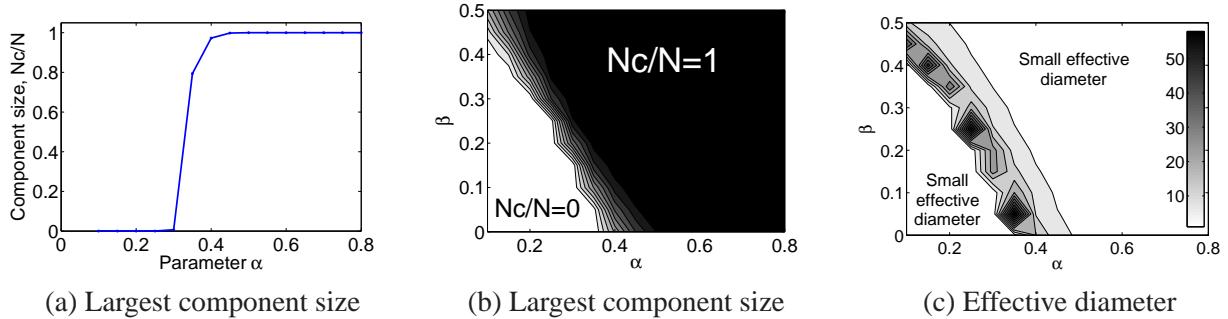


Figure 5.10: Fraction of nodes in the largest weakly connected component (N_c/N) and the effective diameter for 4-star initiator graph. (a) We fix $\beta = 0.15$ and vary α . (b) We vary both α and β . (c) Effective diameter of the network, if network is disconnected or very dense path lengths are short, the diameter is large when the network is barely connected.

$a = 1.05$. Last, we see an example of a graph with shrinking/stabilizing effective diameter. Here we set the $\alpha = 0.54$ which results in a densification exponent of 1.2. Note that these observations are not contradicting Theorem 5.3.12. Actually, these simulations here agree well with the analysis of Mahdian and Xu [Mahdian and Xu, 2007].

Last, we examine the parameter space of a Stochastic Kronecker graph where we choose a star on 4 nodes as a initiator graph and use the familiar parameterization, using α and β . The initiator graph and the structure of the corresponding (deterministic) Kronecker graph adjacency matrix is shown in top row of Figure 5.3.

Figure 5.10(a) shows the sharp transition in the fraction of the number of nodes that belong to the largest weakly connected component as we fix $\beta = 0.15$ and slowly increase α . Such phase transitions on the size of the largest connected component also occur in Erdős-Rényi random graphs. Figure 5.10(b) further explores this by plotting the fraction of nodes in the largest connected component (N_c/N) over the full parameter space. Notice a sharp transition between disconnected (white area) and connected graphs (dark).

Last, Figure 5.10(c) shows the effective diameter over the parameter space (α, β) for the 4-node star initiator graph. Notice that when parameter values are small, the effective diameter is small, since the graph is disconnected and not many pairs of nodes can be reached. The shape of the transition between low-high diameter closely follows the shape of the emergence of the connected component. Similarly, when parameter values are large, graph is very dense, and the diameter is small. There is a narrow band in parameter space where we get graphs with interesting diameters.

5.5 Kronecker graphs model estimation

In previous sections we proved that shapes (parametric forms) of various network properties of Kronecker graphs follow those found in real networks. Moreover, we also gave closed form expressions that allow us to calculate a property (*e.g.*, diameter, eigenvalue spectrum) of a network given just the initiator matrix. So in principle, one could invert the equations and directly get from a property (*e.g.*, shape of degree distribution) to the values of initiator matrix.

However, in previous section we did not say anything about how various network properties of a Kronecker graph correlate and interdepend. For example, it could be the case that they are mutually exclusive. So one could, for instance, only match the network diameter but not the degree distribution or vice versa. However, as we show later this is not the case.

Now we turn our attention to automatically estimating the Kronecker initiator graph. The setting is that we are given a real network G and would like to find a Stochastic Kronecker initiator \mathcal{P}_1 that produces a synthetic Kronecker graph K that is “similar” to G . One way to measure similarity is to compare statistical network properties, like diameter and degree distribution, of graphs G and K .

Comparing statistical properties already suggests a very direct approach to this problem: One could first identify the set of statistics to match, then define an error metric and somehow optimize over it. For example, one could use the KL divergence [Kullback and Leibler, 1951], or the sum of squared differences between the degree distribution of the real network G and its synthetic counterpart K . Moreover, as we are interested in matching several such statistics between the networks one would have to meaningfully combine these individual error metrics into a global error metric. So, one would have to specify what kind of properties he or she cares about and then combine them accordingly. This would be a hard task as the patterns of interest have very different magnitudes and scales. Moreover, as new network patterns are discovered, the error functions would have to be changed and models re-estimated. And even then it is not clear how to define the optimization procedure and how to perform optimization over the parameter space.

Our approach here is different. Instead of committing to a set of network properties ahead of time, we will try to directly match the adjacency matrices of real network G and its synthetic counterpart K . The idea is that if the adjacency matrices are similar then the global statistical properties (statistics computed over K and G) will also match. Moreover, by directly working with the graph itself (and not summary statistics), we do not commit to any particular set of network statistics (network properties/patterns) and as new statistical properties of networks are discovered our models and estimated parameters still hold.

5.5.1 Preliminaries

Stochastic graph models introduce probability distributions over graphs. A generative model assigns a probability $P(G)$ to every graph G . $P(G)$ is the *likelihood* that a given model (with a given set of parameters) generated graph G . We concentrate on Stochastic Kronecker Graph model, and consider fitting it to a real graph G , our data. We use maximum likelihood approach, *i.e.*, we aim to find parameter values, the initiator \mathcal{P}_1 , that maximize the $P(G)$ under the Stochastic Kronecker model.

This presents several challenges:

- **Model selection:** Graph is a single structure, and not a set of items drawn i.i.d. from some distribution. So one can not split it into independent training and test sets. The fitted parameters will thus be best to generate a *particular* instance of a graph. Also, overfitting could be an issue since more complex model generally fits better.
- **Node correspondence:** The second challenge is the node correspondence or node labeling problem. Graph G has a set of N nodes, and each node has unique index (label, id). Labels do not carry any particular meaning, they just uniquely denote or identify the nodes. One can think of this as the graph is first generated and then the labels (node ids) are randomly assigned. This means that two isomorphic graphs that have different node ids should have the same likelihood. Permutation σ is

sufficient to describe the node correspondences as it maps labels (ids) to nodes of the graph. To compute the likelihood $P(G)$ one has to consider all node correspondences $P(G) = \sum_{\sigma} P(G|\sigma)P(\sigma)$, where the sum is over all $N!$ permutations σ of N nodes. Calculating this *super-exponential* sum explicitly is unfeasible for any graph with more than a handful of nodes. Intuitively, one can think of this summation as some kind of graph isomorphism test where we are searching for best correspondence (mapping) between nodes of G and \mathcal{P} .

- **Likelihood estimation:** Calculating $P(G|\sigma)$ naively takes $O(N^2)$ as one has to evaluate the probability of each of the N^2 possible edges in the graph adjacency matrix. Again, for graphs of size we want to model here, approaches with quadratic complexity are infeasible.

To develop our solution we use sampling to avoid super-exponential sum over the node correspondences. By exploiting the structure of the Kronecker matrix multiplication we develop an algorithm to evaluate $P(G|\sigma)$ in *linear* time $O(E)$. Since real graphs are *sparse*, *i.e.*, the number of edges is roughly of the same order as the number of nodes, this makes fitting of Kronecker Graphs to large networks feasible.

5.5.2 Problem formulation

Suppose we are given a graph G on $N = N_1^k$ nodes (for some positive integer k), and a $N_1 \times N_1$ Stochastic Kronecker Graph initiator matrix \mathcal{P}_1 . Here \mathcal{P}_1 is a parameter matrix, a set of parameters that we aim to estimate. For now also assume N_1 , the size of the initiator matrix, is given. Later we will show how to automatically select it. Next, using \mathcal{P}_1 we create a Stochastic Kronecker Graph probability matrix \mathcal{P}_k , where every entry p_{uv} of \mathcal{P}_k contains a probability that node u links to node v . We then evaluate the probability that G is a realization of \mathcal{P}_k . The task is to find such \mathcal{P}_1 that has the highest probability of realizing (generating) G .

Formally, we are solving:

$$\arg \max_{\mathcal{P}_1} P(G|\mathcal{P}_1) \quad (5.3)$$

To keep the notation simpler we use standard symbol Θ to denote the parameter matrix \mathcal{P}_1 that we are trying to estimate. We denote entries of $\Theta = \mathcal{P}_1 = [\theta_{ij}]$, and similarly we denote $\mathcal{P} = \mathcal{P}_k = [p_{ij}]$. Note that here we slightly simplified the notation: we use Θ to refer to \mathcal{P}_1 , and θ_{ij} are elements of Θ . Similarly, p_{ij} are elements of \mathcal{P} ($\equiv \mathcal{P}_k$). Moreover, we denote $K = R(\mathcal{P})$, *i.e.*, K is a realization of the Stochastic Kronecker graph sampled from probabilistic adjacency matrix \mathcal{P} .

As noted before, the node ids are assigned arbitrary and they carry no significant information, which means that we have to consider all the mappings of nodes from G to rows and columns of stochastic adjacency matrix \mathcal{P} . A priori all labelings are equally likely. A permutation σ of the set $\{1, \dots, N\}$ defines this mapping of nodes from G to stochastic adjacency matrix \mathcal{P} . To evaluate the likelihood of G one needs to consider all possible mappings of N nodes of G to rows (columns) of \mathcal{P} . For convenience we work with *log-likelihood* $l(\Theta)$, and solve $\hat{\Theta} = \arg \max_{\Theta} l(\Theta)$, where $l(\Theta)$ is defined as:

$$\begin{aligned} l(\Theta) &= \log P(G|\Theta) = \log \sum_{\sigma} P(G|\Theta, \sigma)P(\sigma|\Theta) \\ &= \log \sum_{\sigma} P(G|\Theta, \sigma)P(\sigma) \end{aligned} \quad (5.4)$$

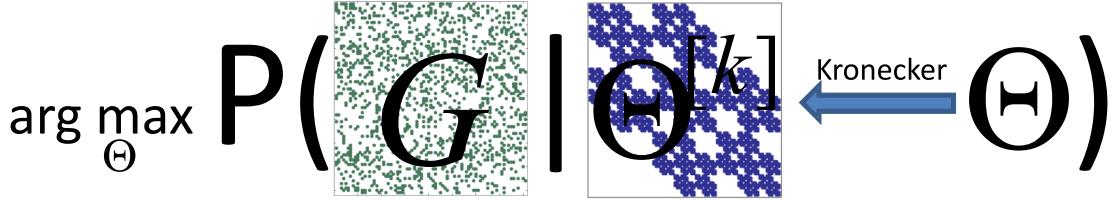


Figure 5.11: Kronecker parameter estimation as an optimization problem. We search over the initiator matrices Θ ($\equiv \mathcal{P}_1$). Using Kronecker multiplication we create probabilistic adjacency matrix $\Theta^{[k]}$ that is of same size as real network G . Now, we evaluate the likelihood by simultaneously traversing and multiplying entries of G and $\Theta^{[k]}$ (see Eq. 5.5). As shown by the figure permutation σ plays an important role, as permuting rows and columns of G could make it look more similar to $\Theta^{[k]}$ and thus increase the likelihood.

The likelihood that a given initiator matrix Θ and permutation σ gave rise to the real graph G , $P(G|\Theta, \sigma)$, is calculated naturally as follows. First, by using Θ we create the Stochastic Kronecker graph adjacency matrix $\mathcal{P} = \mathcal{P}_k = \Theta^{[k]}$. Permutation σ defines the mapping of nodes of G to the rows and columns of stochastic adjacency matrix \mathcal{P} . (See Figure 5.11 for illustration.) We then model edges as independent Bernoulli random variables parameterized by the parameter matrix Θ . So, each entry p_{uv} of \mathcal{P} gives exactly the probability of edge (u, v) appearing.

We then define the likelihood:

$$P(G|\mathcal{P}, \sigma) = \prod_{(u,v) \in G} \mathcal{P}[\sigma_u, \sigma_v] \prod_{(u,v) \notin G} (1 - \mathcal{P}[\sigma_u, \sigma_v]), \quad (5.5)$$

where we denote σ_i as the i^{th} element of the permutation σ , and $\mathcal{P}[i, j]$ is the element at row i , and column j of matrix $\mathcal{P} = \Theta^{[k]}$.

The likelihood is defined very naturally. We traverse the entries of adjacency matrix G and then based on whether a particular edge appeared in G or not we take the probability of edge occurring (or not) as given by \mathcal{P} , and multiply these probabilities. As one has to touch all the entries of the stochastic adjacency matrix \mathcal{P} evaluating Equation 5.5 takes $O(N^2)$.

We further illustrate the process of estimating Stochastic Kronecker initiator matrix Θ in Figure 5.11. We search over initiator matrices Θ to find the one that maximizes the likelihood $P(G|\Theta)$. To estimate $P(G|\Theta)$ we are given a concrete Θ and now we use Kronecker multiplication to create probabilistic adjacency matrix $\Theta^{[k]}$ that is of same size as real network G . Now, we evaluate the likelihood by traversing the corresponding entries of G and $\Theta^{[k]}$. Equation 5.5 basically traverses the adjacency matrix of G , and maps every entry (u, v) of G to a corresponding entry (σ_u, σ_v) of \mathcal{P} . Then in case that edge (u, v) exists in G (i.e., $G[u, v] = 1$) likelihood that particular edge existing is $\mathcal{P}[\sigma_u, \sigma_v]$, and similarly, in case the edge (u, v) does not exist the likelihood is simply $1 - \mathcal{P}[\sigma_u, \sigma_v]$. This also demonstrates the importance of permutation σ , as permuting rows and columns of G could make the adjacency matrix looking more “similar” to $\Theta^{[k]}$, and would increase the likelihood.

So far we showed how to assess the quality (likelihood) of a particular Θ . So, naively one could perform some kind of exhaustive grid search to find best Θ . However, this is very inefficient. A better way of doing it is to compute the gradient of the log-likelihood $\frac{\partial}{\partial \hat{\Theta}} l(\hat{\Theta})$, and then use the gradient to update the

Algorithm 5.1: KRONFIT algorithm.

input : size of parameter matrix N_1 , graph G on $N = N_1^k$ nodes, and learning rate λ
output: MLE parameters $\hat{\Theta}$ ($N_1 \times N_1$ probability matrix)

```

initialize  $\hat{\Theta}_1$ 
while not converged do
    evaluate gradient:  $\frac{\partial}{\partial \hat{\Theta}_t} l(\hat{\Theta}_t)$ 
    update parameter estimates:  $\hat{\Theta}_{t+1} = \hat{\Theta}_t + \lambda \frac{\partial}{\partial \hat{\Theta}_t} l(\hat{\Theta}_t)$ 
end
return  $\hat{\Theta} = \hat{\Theta}_t$ 
```

current estimate of Θ and move towards a solution of higher likelihood. Algorithm 5.1 gives an outline of the optimization procedure.

However, there are several difficulties with this algorithm. First, we are assuming gradient descent type optimization will work, *i.e.* the problem does not have (too many) local minima. Second, we are summing over exponentially many permutations in equation 5.4. Third, the evaluation of equation 5.5 as it is written takes $O(N^2)$ and needs to be evaluated $N!$ times. So, just naively calculating the likelihood takes $O(N!N^2)$.

Observation 5.5.1. *The complexity of calculating the likelihood $P(G|\Theta)$ of the graph G naively is $O(N!N^2)$, where N is the number of nodes in G .*

Next, we show that all this can be done in *linear time*.

5.5.3 Summing over the node labelings

To maximize equation 5.3 using algorithm 5.1 we need to obtain the gradient of the log-likelihood $\frac{\partial}{\partial \Theta} l(\Theta)$. We can write:

$$\begin{aligned}
\frac{\partial}{\partial \Theta} l(\Theta) &= \frac{\sum_{\sigma} \frac{\partial}{\partial \Theta} P(G|\sigma, \Theta) P(\sigma)}{\sum_{\sigma'} P(G|\sigma', \Theta) P(\sigma')} \\
&= \frac{\sum_{\sigma} \frac{\partial \log P(G|\sigma, \Theta)}{\partial \Theta} P(G|\sigma, \Theta) P(\sigma)}{P(G|\Theta)} \\
&= \sum_{\sigma} \frac{\partial \log P(G|\sigma, \Theta)}{\partial \Theta} P(\sigma|G, \Theta)
\end{aligned} \tag{5.6}$$

Note we are still summing over all $N!$ permutations σ , so calculating eq. 5.6 is computationally intractable for graphs with more than a handful of nodes. However, the equation has a nice form which allows for use of simulation techniques to avoid the summation over super-exponentially many node correspondences. Thus, we simulate draws from the permutation distribution $P(\sigma|G, \Theta)$, and then evaluate the quantities at the sampled permutations to obtain the expected values of log-likelihood and gradient. Algorithm 5.2 gives the details.

Algorithm 5.2: Calculating log-likelihood and gradient

input : Parameter matrix Θ , and graph G
output: Log-likelihood $l(\Theta)$, and gradient $\frac{\partial}{\partial \Theta} l(\Theta)$

```

for  $t := 1$  to  $T$  do
     $\sigma_t := \text{SamplePermutation}(G, \Theta)$ 
     $l_t = \log P(G|\sigma^{(t)}, \Theta)$ 
     $grad_t := \frac{\partial}{\partial \Theta} \log P(G|\sigma^{(t)}, \Theta)$ 
end
return  $l(\Theta) = \frac{1}{T} \sum_t l_t$ , and  $\frac{\partial}{\partial \Theta} l(\Theta) = \frac{1}{T} \sum_t grad_t$ 

```

Sampling permutations

Next, we describe the Metropolis algorithm to simulate draws from the permutation distribution $P(\sigma|G, \Theta)$, which is given by

$$P(\sigma|G, \Theta) = \frac{P(\sigma, G, \Theta)}{\sum_{\sigma} P(\sigma, G, \Theta)} = \frac{\sum_{\sigma} P(\sigma, G, \Theta)}{Z_{\sigma}}$$

where Z_{σ} is the normalizing constant that is hard to compute since it involves the sum over $N!$ elements. However, if we compute the likelihood ratio between permutations σ and σ' (Equation 5.7) the normalizing constants nicely cancel out:

$$\frac{P(\sigma'|G, \Theta)}{P(\sigma|G, \Theta)} = \prod_{(u,v) \in G} \frac{\mathcal{P}[\sigma_u, \sigma_v]}{\mathcal{P}[\sigma'_u, \sigma'_v]} \prod_{(u,v) \notin G} \frac{(1 - \mathcal{P}[\sigma_u, \sigma_v])}{(1 - \mathcal{P}[\sigma'_u, \sigma'_v])} \quad (5.7)$$

$$= \prod_{\substack{(u,v) \in G \\ (\sigma_u, \sigma_v) \neq (\sigma'_u, \sigma'_v)}} \frac{\mathcal{P}[\sigma_u, \sigma_v]}{\mathcal{P}[\sigma'_u, \sigma'_v]} \prod_{\substack{(u,v) \notin G \\ (\sigma_u, \sigma_v) \neq (\sigma'_u, \sigma'_v)}} \frac{(1 - \mathcal{P}[\sigma_u, \sigma_v])}{(1 - \mathcal{P}[\sigma'_u, \sigma'_v])} \quad (5.8)$$

This immediately suggests to use of Metropolis sampling algorithm [Gamerman, 1997] to simulate draws from the permutation distribution since Metropolis is solely based on such ratios (where normalizing constants cancel out). In particular, suppose that in the Metropolis algorithm (Algorithm 5.3) we consider a move from permutation σ to a new permutation σ' . Probability of accepting the move to σ' is given by Equation 5.7 (if $\frac{P(\sigma'|G, \Theta)}{P(\sigma|G, \Theta)} \leq 1$) or 1 otherwise.

Now we have to devise a way to sample permutations σ from the proposal distribution. One way to do this would be to simply generate a random permutation σ' and then check the acceptance condition. This would be very inefficient as we expect the distribution $P(\sigma|G, \Theta)$ to be heavily skewed, *i.e.*, there will be a relatively small number of good node mappings. Even more so as the degree distributions in real networks are skewed there will be many bad permutations with low likelihood, and few good ones that do a good job in matching nodes of high degree.

To make the sampling process “smoother”, *i.e.*, sample permutations that are not that different (and thus are not randomly jumping across the permutation space) we design a Markov chain. The idea is to stay in high likelihood part of permutation space longer. We do this by making samples dependent, *i.e.*, given σ' we want to generate next candidate permutation σ'' to then evaluate the likelihood ratio. When designing

Algorithm 5.3: SamplePermutation(G, Θ): Metropolis sampling of the node permutation.

input : Kronecker initiator matrix Θ and a graph G on N nodes

output: Permutation $\sigma^{(i)} \sim P(\sigma|G, \Theta)$

$\sigma^{(0)} := (1, \dots, N)$

$i = 1$

repeat

 Draw j and k uniformly from $(1, \dots, N)$

$\sigma^{(i)} := \text{SwapNodes}(\sigma^{(i-1)}, j, k)$

 Draw u from $U(0, 1)$

if $u > \frac{P(\sigma^{(i)}|G, \Theta)}{P(\sigma^{(i-1)}|G, \Theta)}$ **then**

$\sigma^{(i)} := \sigma^{(i-1)}$

end

$i = i + 1$

until $\sigma^{(i)} \sim P(\sigma|G, \Theta)$

return $\sigma^{(i)}$

Where $U(0, 1)$ is a uniform distribution on $[0, 1]$, and $\sigma' := \text{SwapNodes}(\sigma, j, k)$ is the permutation σ' obtained from σ by swapping elements at positions j and k .

the Markov chain step one has to be careful so that the proposal distribution satisfies the detailed balance condition. This means that probability of a generating a candidate σ'' from σ' has to be same as transition in the opposite way, $P(\sigma' \rightarrow \sigma'') = P(\sigma'' \rightarrow \sigma')$.

In algorithm 5.3 we use a simple proposal where given permutation σ' we generate σ'' by swapping elements at two uniformly at random chosen positions of σ' . We refer to this proposal as SwapNodes. While this is simple and clearly satisfies the detailed balance condition it is also inefficient in a way that most of the times low degree nodes will get swapped (a direct consequence of heavy tailed degree distributions). This has two consequences, (a) we will slowly converge to good permutations (accurate mappings of high degree nodes), and (b) once we reach a good permutation, very few permutations will get accepted as most proposed permutations σ' will swap low degree nodes (as they form the majority of nodes).

A possibly more efficient way would be to swap elements of σ biased based on corresponding node degree. However, doing this directly does not satisfy the detailed balance condition. A way of sampling labels biased by node degrees that at the same time satisfies the detailed balance condition is the following: we pick an edge in G uniformly at random and swap the labels of the endpoints. Notice this is biased towards swapping labels of nodes with high degrees simply as they have more edges. The detailed balance condition holds as edges are sampled uniformly at random. We refer to this proposal as SwapEdgeEndpoints.

However, the issue with this proposal is that if the graph G is disconnected, we will only be swapping labels of nodes that belong to the same connected component. This means that some parts of the permutation space will never get visited. To overcome this problem we execute SwapNodes with some probability ω and SwapEdgeEndpoints with probability $1 - \omega$.

To summarize we consider the following two permutation proposal distributions:

- $\sigma'' = \text{SwapNodes}(\sigma')$: we obtain σ'' by taking σ' , uniformly at random selecting a pair of elements

and swapping their positions.

- $\sigma'' = \text{SwapEdgeEndpoints}(\sigma')$: we obtain σ'' from σ' by first sampling an edge (j, k) from G uniformly at random, then we take σ' and swap the labels at positions j and k .

Speeding up the likelihood ratio calculation

We further speed up the algorithm by using the following observation. As written the equation 5.7 takes $O(N^2)$ to evaluate since we have to consider N^2 possible edges. However, notice that permutations σ and σ' differ only at two positions, *i.e.* elements at position j and k are swapped, *i.e.*, σ and σ' map all nodes except the two to the same locations. This means those elements of equation 5.7 cancel out. Thus to update the likelihood we only need to traverse two rows and columns of matrix \mathcal{P} , namely rows and columns j and k , since everywhere else the mapping of the nodes to the adjacency matrix is the same for both permutations. This gives equation 5.8 where the products now range only over the two rows/columns of \mathcal{P} where σ and σ' differ.

Graphs we are working with here are too large to allow us to explicitly create and store the stochastic adjacency matrix \mathcal{P} by Kronecker powering the initiator matrix Θ . Every time probability $\mathcal{P}[i, j]$ of edge (i, j) is needed the equation 5.2 is evaluated, which takes $O(k)$. So a single iteration of algorithm 5.3 takes $O(kN)$.

Observation 5.5.2. *Sampling a permutation σ from $P(\sigma|G, \Theta)$ takes $O(kN)$.*

This gives us an improvement over the $O(N!)$ complexity of summing over all the permutations. So far we have shown how to obtain a permutation but we still need to evaluate the likelihood and find the gradients that will guide us in finding good initiator matrix. The problem here is that naively evaluating the network likelihood (gradient) as written in equation 5.6 takes time $O(N^2)$. This is exactly what we investigate next and how to calculate the likelihood in *linear time*.

5.5.4 Efficiently evaluating likelihood and gradient

We just showed how to efficiently sample node permutations. Now, given a permutation we show how to efficiently evaluate the likelihood and its gradient. Similarly as evaluating the likelihood ratio, naively calculating the log-likelihood $l(\Theta)$ or its gradient $\frac{\partial}{\partial \Theta} l(\Theta)$ takes time quadratic in the number of nodes. Next, we show how to compute this in linear time $O(E)$.

We begin with the observation that real graphs are sparse, which means that the number of edges is not quadratic but rather almost linear in the number of nodes, $E \ll N^2$. This means that majority of entries of graph adjacency matrix are zero, *i.e.*, most of the edges are not present. We exploit this fact. The idea is to first calculate the likelihood (gradient) of an empty graph, *i.e.*, a graph with zero edges, and then correct for the edges that actually appear in G .

To naively calculate the likelihood for an empty graph one needs to evaluate every cell of graph adjacency matrix. We consider Taylor approximation to the likelihood, and exploit the structure of matrix \mathcal{P} to devise a constant time algorithm.

First, consider the second order Taylor approximation to log-likelihood of an edge that succeeds with

probability x but does not appear in the graph:

$$\log(1 - x) \approx -x - \frac{1}{2}x^2$$

Calculating $l_e(\Theta)$, the log-likelihood of an empty graph, becomes:

$$l_e(\Theta) = \sum_{i=1}^N \sum_{j=1}^N \log(1 - p_{ij}) \approx - \left(\sum_{i=1}^{N_1} \sum_{j=1}^{N_1} \theta_{ij} \right)^k - \frac{1}{2} \left(\sum_{i=1}^{N_1} \sum_{j=1}^{N_1} \theta_{ij}^2 \right)^k \quad (5.9)$$

Notice that while the first pair of sums ranges over N elements, the last pair only ranges over N_1 elements ($N_1 = \log_k N$). Equation 5.9 holds due to the recursive structure of matrix \mathcal{P} generated by the Kronecker product. We substitute the $\log(1 - p_{ij})$ with its Taylor approximation, which gives a sum over elements of \mathcal{P} and their squares. Next, we notice the sum of elements of \mathcal{P} forms a multinomial series, and thus $\sum_{i,j} p_{ij} = (\sum_{i,j} \theta_{ij})^k$, where θ_{ij} denotes an element of Θ , and p_{ij} element of $\Theta^{[k]}$.

Calculating log-likelihood of G now takes $O(E)$: First, we calculate the likelihood of an empty graph in constant time, and then account for the edges that are actually present in G , *i.e.*, we subtract no-edge likelihood and add the edge likelihoods:

$$l(\Theta) = l_e(\Theta) + \sum_{(u,v) \in G} -\log(1 - \mathcal{P}[\sigma_u, \sigma_v]) + \log(\mathcal{P}[\sigma_u, \sigma_v])$$

5.5.5 Calculating the gradient

Calculation of the gradient of log-likelihood follows exactly the same pattern as described above. We first calculate gradient as if graph G would have no edges. Then we correct the gradient for the edges that are present in G . As in previous section we speed up the calculations of the gradient by exploiting the fact that two consecutive permutations σ and σ' differ only at two positions, and thus given the gradient from previous step one only needs to account for the swap of the two rows and columns of the gradient matrix $\partial \mathcal{P} / \partial \Theta$ to update to the gradients of individual parameters.

5.5.6 Determining the size of initiator matrix

The question we answer next is how to determine the right number of parameters, *i.e.*, what is the right size of Θ matrix? This is a classical question of model selection where there is a tradeoff between the complexity of the model, and the quality of the fit. Bigger model with more parameters usually fits better, however it is also more likely to overfit the data.

For model selection to find the appropriate value of N_1 , the size of matrix Θ , and choose the right tradeoff between the complexity of the model and the quality of the fit, we propose to use the Bayes Information Criterion (BIC) [Schwarz, 1978]. Stochastic Kronecker Graphs model the presence of edges with independent Bernoulli random variables, where the canonical number of parameters is N_1^{2k} , which is a function of a lower-dimensional parameter Θ . This is then a *curved exponential family* [Efron, 1975], and BIC naturally applies:

$$\text{BIC} = -l(\hat{\Theta}) + \frac{1}{2} N_1^2 \log(N^2)$$

where $\hat{\Theta}$ are maximum likelihood parameters under the model with $\hat{\Theta}$ of size $N_1 \times N_1$, and N is the number of nodes in G .

Similarly, to BIC one could also consider the Minimum Description Length (MDL) [Rissanen, 1978] principle where the model is scored by the quality of the fit plus the size of the description that encodes the model and the parameters.

5.6 Experiments on real and synthetic data

We divide the experiments into several subsections. First we examine the convergence and mixing of the Markov chain of our permutation sampling scheme. Then we consider estimating the parameters of the synthetic Kronecker graphs to see whether KRONFIT is able to recover the parameters used to generate the network. Last, we consider fitting Stochastic Kronecker Graph to large real world networks.

5.6.1 Permutation sampling

In our experiments we considered both synthetic and real graphs. Unless mentioned otherwise all synthetic Kronecker graphs were generated using $\mathcal{P}_1^* = [0.8, 0.6; 0.5, 0.3]$, and $k = 14$ which gives us a graph G on $N = 16,384$ nodes and $E = 115,741$ edges. We chose this particular \mathcal{P}_1^* as it closely resembles a typical initiator for real networks (that we show later).

Convergence of the log-likelihood and the gradient

First, we examine the convergence of Metropolis permutation sampling. As every next permutation is obtained from the previous one by locally modifying it this creates a Markov chain. We want to assess the convergence and mixing of the chain, *i.e.*, determine how many permutations one needs to draw to reliably estimate the likelihood and the gradient, and also how long does it take till the samples converge to the stationary distribution. For the experiment we generated a synthetic Stochastic Kronecker Graph using \mathcal{P}_1^* as defined above. Then, starting with a random permutation we run algorithm 5.3, and measure how the likelihood and the gradients converge to their true values.

In this particular case we first generated Stochastic Kronecker Graph G as described above, but then calculated the likelihood and the parameter gradients for $\Theta' = [0.8, 0.75; 0.45, 0.3]$. We average the likelihoods and gradients over buckets of 1,000 consecutive samples, and plot how the log-likelihood calculated over the sampled permutations approaches the true log-likelihood (that we can compute since G is a Stochastic Kronecker Graph).

First, we present experiments that aim to answer how many samples (*i.e.*, permutations) does one need to draw to obtain a reliable estimate of the gradient (see Equation 5.6). Figure 5.12(a) shows how the estimated log-likelihood approaches the true likelihood. Notice that estimated values quickly converge to their true values, *i.e.*, Metropolis sampling quickly moves towards “good” permutations. Similarly, Figure 5.12(b) plots the convergence of the gradients. Notice that θ_{11} and θ_{22} of Θ' and \mathcal{P}_1^* match, so gradients of these two parameters should converge to zero and indeed they do. On the other hand, θ_{12} and θ_{21} differ between Θ' and \mathcal{P}_1^* . Notice, the gradient for one is positive as the parameter θ_{12} of Θ' should be decreased, and similarly for θ_{21} the gradient is negative as the parameter value should be increased to

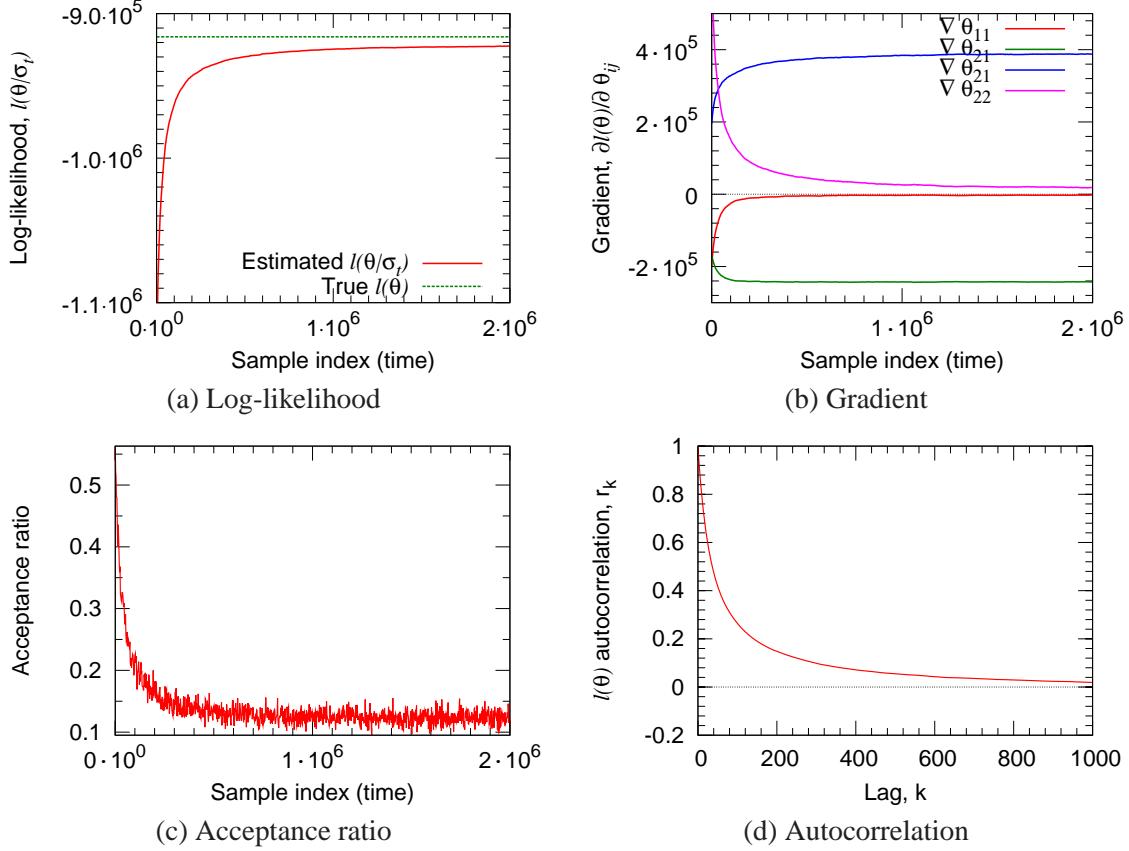


Figure 5.12: Convergence of the log-likelihood and gradients towards their true values for Metropolis permutation sampling (algorithm 5.3) with the number of samples.

match the Θ' . In summary, this shows that log-likelihood and gradients rather quickly converge to their true values.

Moreover, in Figures 5.12(c) and (d) we investigate the properties of the Markov Chain Monte Carlo sampling procedure, and asses convergence and mixing criteria. First, we plot the fraction of accepted proposals. It stabilizes at around 15%, which is quite close to the rule-of-a-thumb of 25%. Second, Figure 5.12(d) plots the autocorrelation of the log-likelihood as a function of the lag. Autocorrelation r_k of a signal X is a function of the lag k where r_k is defined as the correlation of signal X at time t with X at $t + k$, *i.e.*, correlation of the signal with itself at lag k . High autocorrelations within chains indicate slow mixing and, usually, slow convergence. On the other hand fast decay of autocorrelation means better the mixing and thus one needs less samples to accurately estimate the gradient or the likelihood. Notice rather fast autocorrelation decay.

All in all, these experiments show that one needs to sample an order of tens of thousands of permutations for the estimates to converge. We also verified that the variance of the estimates is sufficiently small. In our experiments we start with a random permutation and use long burn-in time. Then when performing optimization we use the permutation from previous step to initialize the permutation at current step of gradient descent. The intuition is that small changes in $P(\sigma|G, \Theta)$ also mean small changes in Θ .

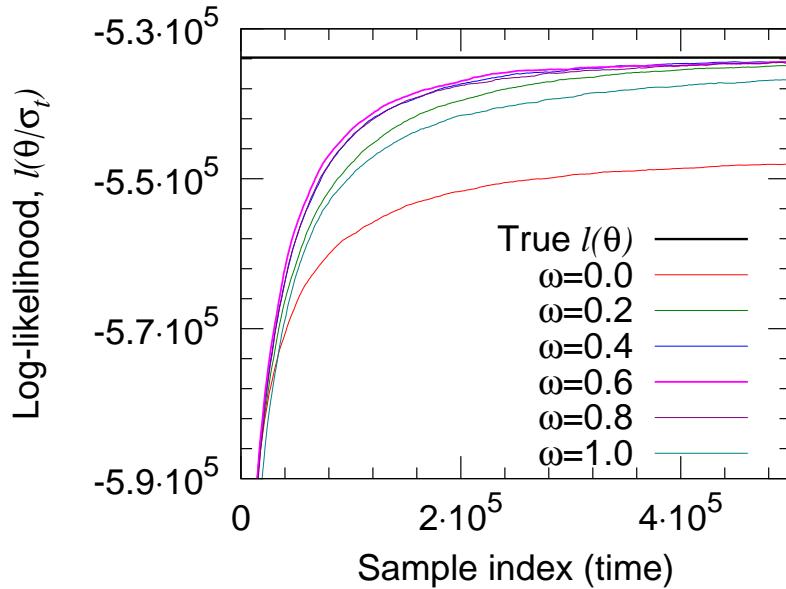


Figure 5.13: Convergence of the log-likelihood and gradients for Metropolis permutation sampling (algorithm 5.3) for different choices of ω that interpolates between the SwapNodes ($\omega = 1$) and SwapEdgeEndpoints ($\omega = 0$) permutation proposal distributions.

Different proposal distributions

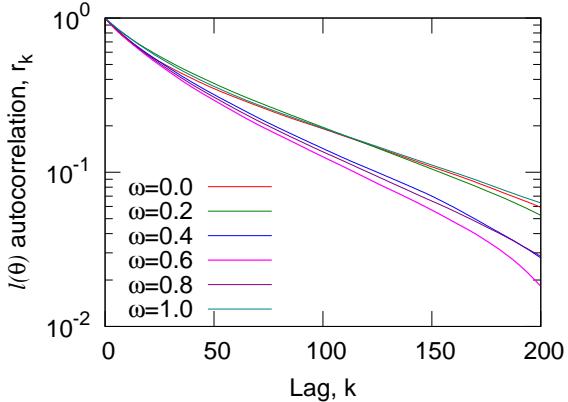
In section 5.5.3 we defined two permutation sampling proposal distributions: SwapNodes where we pick two nodes uniformly at random and swap their labels (node ids); and SwapEdgeEndpoints where we pick a random edge in a graph and then swap the labels of the edge endpoints. We also discussed that one can interpolate between the two strategies by executing SwapNodes with probability ω , and SwapEdgeEndpoints with probability $1 - \omega$.

So, given a Stochastic Kronecker Graph G on $N = 16,384$ and $E = 115,741$ generated from $\mathcal{P}_1^* = [0.8, 0.7; 0.5, 0.3]$ we evaluate the likelihood of $\Theta' = [0.8, 0.75; 0.45, 0.3]$. As we sample permutations we observe how the estimated likelihood converges to the true likelihood. Moreover we also vary parameter ω that interpolates between the two permutation proposal distributions. The quicker the converge towards the true log-likelihood the better the proposal distribution.

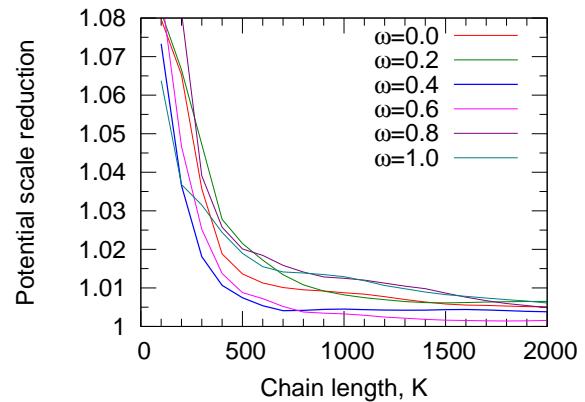
Figure 5.13 plots the convergence of the log-likelihood with the number of sampled permutations. We plot the average over non-overlapping buckets of 1,000 consecutive permutations. Faster convergence means better permutation proposal distribution. When we use only SwapNodes ($\omega = 1$) or SwapEdgeEndpoints ($\omega = 0$) convergence is rather slow. We obtain best convergence for ω around 0.6.

Similarly, Figure 5.14(a) plots the autocorrelation as a function of the lag k for different choices of ω . Faster autocorrelation decay means better mixing of the Markov chain. Again, notice that we get best mixing for $\omega = 0.6$. (Notice logarithmic y-axis.)

Last, we diagnose how long the sampling procedure must be run before the generated samples can be considered to be drawn (approximately) from the stationary distribution. We call this the burn-in time of the chain. There are various procedures for assessing convergence. Here we adopt the approach of Gelman *et al.* [Gelman et al., 2003], that is based on running multiple Markov chains each from a different starting



(a) Autocorrelation



(b) Potential scale reduction

Figure 5.14: (a) Autocorrelation plot of the log-likelihood for the different choices of parameter ω . Notice we get best mixing with $\omega = 0.6$. (b) The potential scale reduction that compares the variance inside- and across- independent Markov chains for different values of parameter ω .

point, and then comparing the variance within the chain and between the chains. The sooner the within- and between-chain variances become equal the faster the burn-in time, *i.e.*, the sooner the samples are drawn from the stationary distribution.

Let l be the parameter that is being simulated with J different chains, and then let $l_j^{(k)}$ denote the k^{th} sample of the j^{th} chain, where $j = 1, \dots, J$ and $k = 1, \dots, K$. More specifically, in our case we run separate permutation sampling chains. So, we first sample permutation $\sigma_j^{(k)}$ and then calculate the corresponding log-likelihood $l_j^{(k)}$.

First, we compute between and within chain variances $\hat{\sigma}_B^2$ and $\hat{\sigma}_W^2$, where between-chain variance is obtained by

$$\hat{\sigma}_B^2 = \frac{K}{J-1} \sum_{j=1}^J (\bar{l}_{\cdot j} - \bar{l}_{\cdot \cdot})^2$$

where $\bar{l}_{\cdot j} = \frac{1}{K} \sum_{k=1}^K l_j^{(k)}$ and $\bar{l}_{\cdot \cdot} = \frac{1}{J} \sum_{j=1}^J \bar{l}_{\cdot j}$

Similarly the within-chain variance is defined by

$$\hat{\sigma}_W^2 = \frac{1}{J(K-1)} \sum_{j=1}^J \sum_{k=1}^K (l_j^{(k)} - \bar{l}_{\cdot j})^2$$

Then, the marginal posterior variance of \hat{l} is calculated using

$$\hat{\sigma}^2 = \frac{K-1}{K} \hat{\sigma}_W^2 + \frac{1}{K} \hat{\sigma}_B^2$$

And, finally, we estimate the *potential scale reduction* [Gelman et al., 2003] of l by

$$\sqrt{\hat{R}} = \sqrt{\frac{\hat{\sigma}^2}{\hat{\sigma}_W^2}}$$

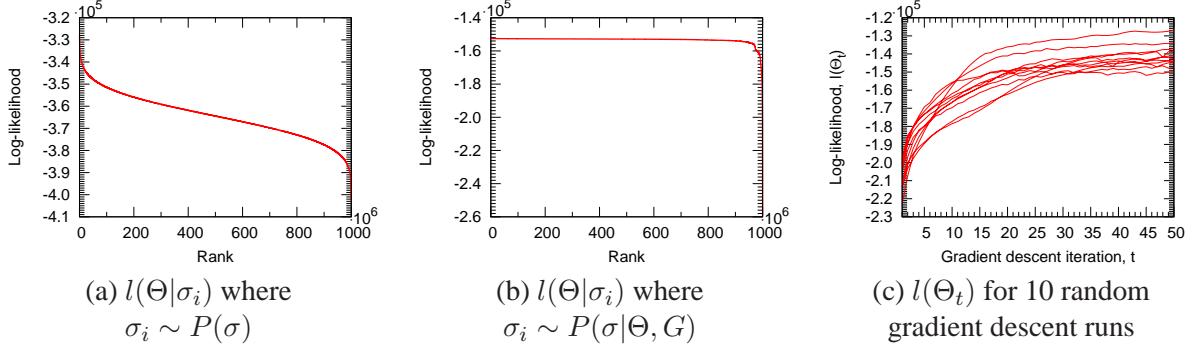


Figure 5.15: (a) Distribution of log-likelihood of permutations sampled uniformly at random, and (b) when sampled from $P(\sigma|\Theta, G)$. Notice the space of good permutations is rather small but our sampling quickly finds permutations of high likelihood. (c) Convergence of log-likelihood for 10 runs of gradient descent, each from a different random starting point.

Note that as the length of the chain $K \rightarrow \infty$ $\sqrt{\hat{R}}$ converges to 1 from above. A recommendation for convergence assessment from [Gelman et al., 2003] is that potential scale reduction is below 1.2.

Figure 5.14(b) gives the Gelman-Rubin-Brooks plot, where we plot the potential scale reduction $\sqrt{\hat{R}}$ over the increasing chain length K for different choices of parameter ω . Notice that the potential scale reduction quickly decays towards 1. Similarly as in Figure 5.14 the extreme values of ω give slow decay, while we obtain fastest potential scale reduction when $\omega \approx 0.6$.

Properties of the permutation space

Next we explore the properties of the permutation space. We would like to quantify what fraction of permutations are “good” (have high likelihood), and how quickly do we discover them. For the experiment we took a real network G (As-ROUTEVIEWS network) and the MLE parameters $\hat{\Theta}$ for it that we estimated before hand ($l(\hat{\Theta}) \approx -150,000$). The network G has 6,474 nodes which means the space of all permutations has $\approx 10^{22,000}$ elements.

First, we sampled 1 billion (10^9) permutations σ_i uniformly at random, *i.e.*, $P(\sigma_i) = 1/(6,474!)$ and for each evaluated its log-likelihood $l(\sigma|\Theta_i) = \log P(\Theta_i|G, \sigma)$. We ordered the permutations in decreasing log-likelihood and plotted $l(\sigma|\Theta_i)$ vs. rank. Figure 5.15(a) gives the plot. Notice that very few random permutations are very bad (*i.e.*, they give low likelihood), similarly few permutations are very good, while most of them are somewhere in between. Notice that best “random” permutation has log-likelihood of $\approx -320,000$, which is far below true likelihood $l(\hat{\Theta}) \approx -150,000$. This suggests that only a very small fraction of all permutations gives good node labelings.

On the other hand, we also repeated the same experiment but now sampled permutations from the permutation distribution $\sigma_i \sim P(\sigma|\Theta, G)$ using our Metropolis sampling scheme. Figure 5.15(b) gives the plot. Notice the radical difference. Now the $l(\sigma|\Theta_i)$ very quickly converges to the true likelihood of $\approx -150,000$. This suggest that while the number of “good” permutations (accurate node mappings) is rather small, our sampling procedure quickly converges to the “good” part of the permutation space where node mappings are accurate.

5.6.2 Properties of the optimization space

In maximizing the likelihood we use stochastic approximation to the gradient. This adds variance to the gradient and makes efficient optimization techniques, like conjugate gradient, highly unstable. Thus we use gradient descent, which is slower but easier to control. First, we make the following observation:

Observation 5.6.1. *Given a real graph G then finding the maximum likelihood Stochastic Kronecker initiator matrix $\hat{\Theta}$*

$$\hat{\Theta} = \arg \max_{\Theta} P(G|\Theta)$$

is a non-convex optimization problem.

Proof. By definition permutations of the Kronecker graphs initiator matrix $\hat{\Theta}$ all have the same log-likelihood. This means that we have several global minima that correspond to permutations of parameter matrix $\hat{\Theta}$, and then between them the log-likelihood drops. This means that the optimization problem is non-convex. \square

The above observation seem not to give much promise to estimating $\hat{\Theta}$ using gradient descent as it is prone to local minima. To check for the presence of other local minima where gradient descent could get stuck we run the following experiment: we generated 100 synthetic Kronecker graphs on 16,384 (2^{14}) nodes and 1.4 million edges on the average, with a randomly chosen 2×2 parameter matrix Θ^* . For each of the 100 graphs we run gradient descent starting from a different random parameter matrix Θ' , and try to recover Θ^* . In 98% of the cases the gradient descent converged to the true parameters. Many times the algorithm converged to a different global minima, *i.e.*, $\hat{\Theta}$ is a permuted version of original parameter matrix Θ^* . Moreover, the median number of gradient descent iterations was only 52.

This suggests surprisingly nice structure of our optimization space: it seems to behave like a convex optimization problem with many equivalent global minima. Moreover, this experiment is also a good sanity check as it shows that given a Kronecker graph we can recover and identify the parameters that were used to generate it.

Moreover, Figure 5.15(c) plots the log-likelihood $l(\Theta_t)$ of the current parameter estimate Θ_t over the iterations t of the stochastic gradient descent. We plot the log-likelihood for 10 different runs of gradient descent, each time starting from a different random set of parameters Θ_0 . Notice that in all runs gradient descent always converges towards the optimum, and none of the runs gets stuck in some local maxima.

5.6.3 Convergence of the graph properties

We approached the problem of estimating Stochastic Kronecker initiator matrix Θ by defining the likelihood over the individual entries of the graph adjacency matrix. However, what we would really like is to be given a real graph G and then generate a synthetic graph K that has similar network properties as G . By properties we mean network statistics that can be computed from the graph, *e.g.*, diameter, degree distribution, clustering coefficient, etc. A priori it is not clear that our approach which tries to match individual entries of graph adjacency matrix will also be able to reproduce these global network statistics. However, as show next this is not the case.

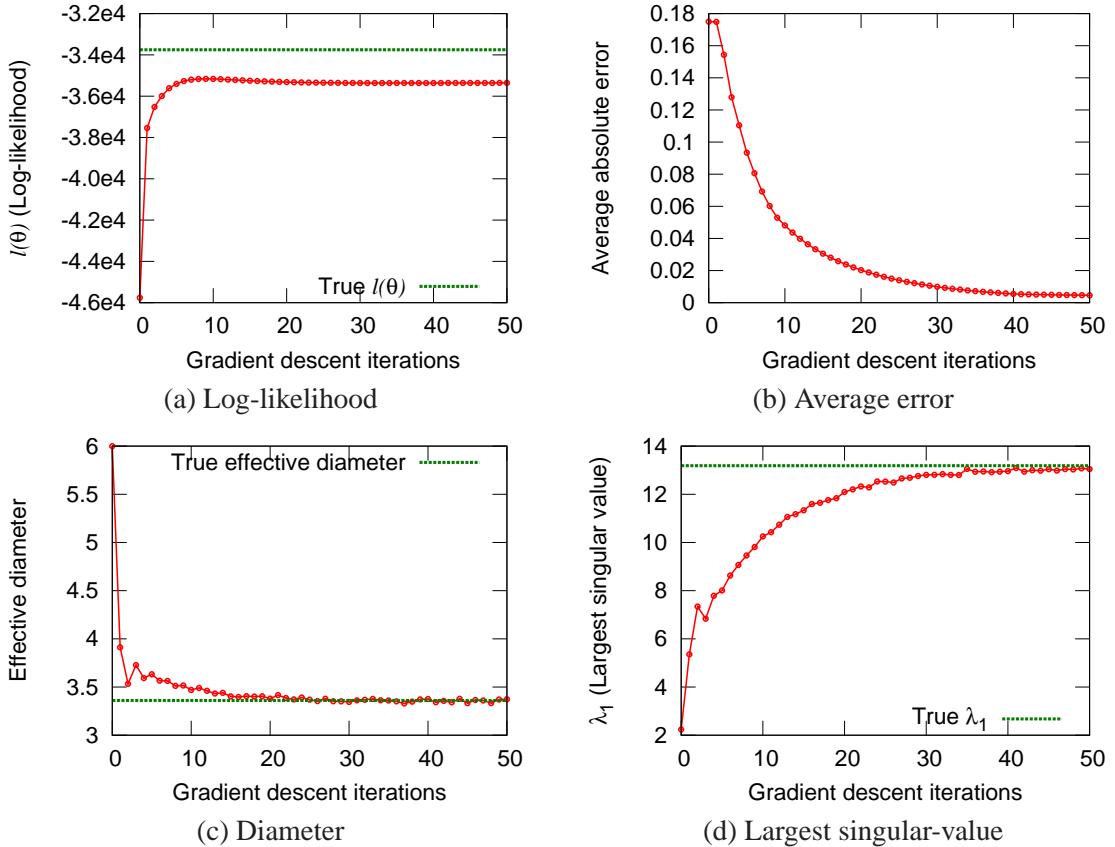


Figure 5.16: Convergence of graph patterns with the number of iterations of gradient descent using the synthetic dataset.

To get some understanding of the convergence of the gradient descent in terms of the network properties we performed the following experiment. After every step t of stochastic gradient descent, we compare the true graph G with the synthetic Kronecker graph K_t generated using the current parameter estimates $\hat{\Theta}_t$. Figure 5.16(a) gives the convergence of log-likelihood, and (b) gives absolute error in parameter values ($\sum |\hat{\theta}_{ij} - \theta_{ij}^*|$, where $\hat{\theta}_{ij} \in \hat{\Theta}_t$, and $\theta_{ij}^* \in \Theta^*$). Similarly, Figure 5.16(c) plots the effective diameter, and (d) gives the largest singular value of graph adjacency matrix K as it converges to largest singular value of G .

Note how with progressing iterations of gradient descent properties of graph K_t quickly converge to those of G even though we are not directly optimizing the similarity in network properties: log-likelihood increases, absolute error of parameters decreases, diameter and largest singular value of K_t both converge to G . This is a nice result as it shows that through maximizing the likelihood the resulting graphs become more and more similar also in their structural properties (even though we are not directly optimizing over them).

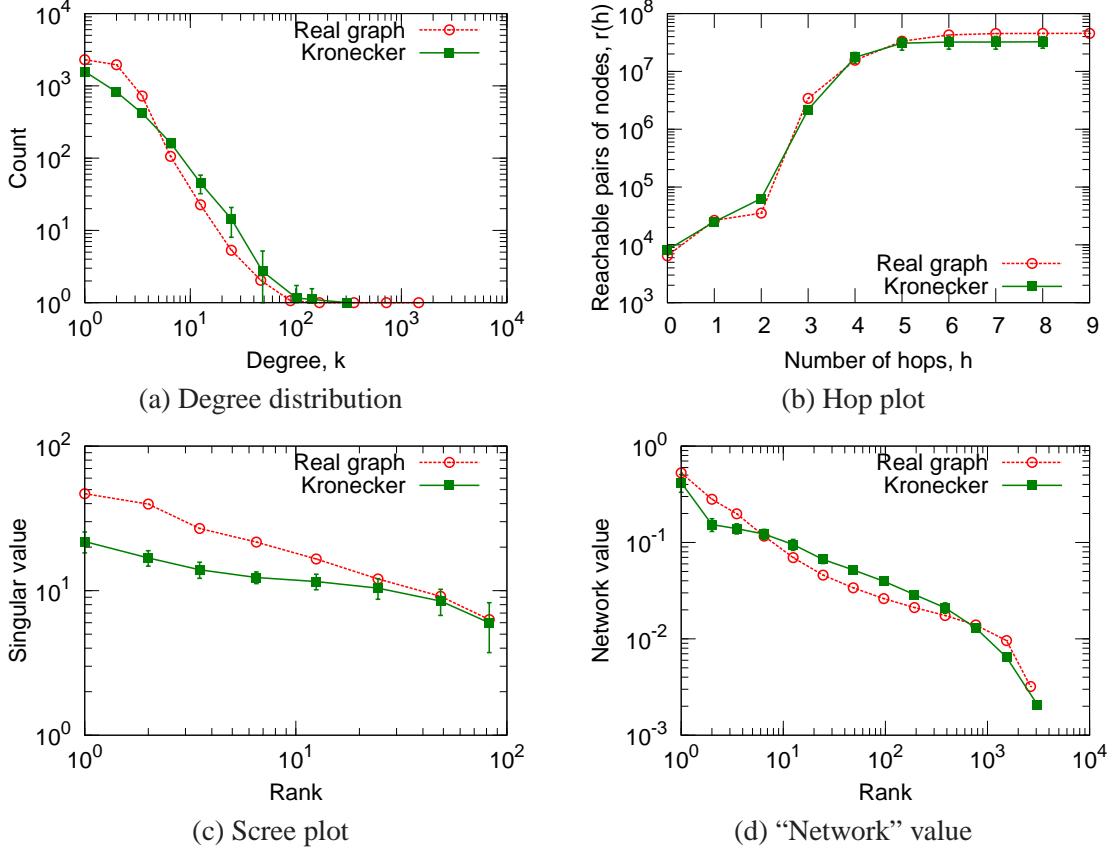


Figure 5.17: Autonomous Systems (AS-ROUTEVIEWS): Overlayed patterns of real graph and the fitted Kronecker graph. Notice that the fitted Kronecker graph matches patterns of the real graph while using only four parameters (2×2 initiator matrix).

5.6.4 Fitting to real-world networks

Next, we present experiments of fitting Kronecker Graphs model to real-world networks. Given a real network G we aim to discover the most likely parameters $\hat{\Theta}$ that ideally would generate a synthetic graph K having similar properties as real G . This assumes that Kronecker Graphs is a good model of the network structure, and that KRONFIT is able to find good parameters. In previous section we showed that KRONFIT can efficiently recover the parameters. Now we examine how well can Kronecker graphs model the structure of real networks.

We consider several different networks, like a graph of connectivity among Internet Autonomous systems (AS-ROUTEVIEWS) with $N = 6,474$ and $E = 26,467$; a who-trusts-whom type social network from Epinions [Richardson et al., 2003] (EPINIONS) with $N = 75,879$ and $E = 508,960$ and many others. The largest network we consider for fitting is FLICKR photo-sharing online social network with 584,207 nodes and 3,555,115 edges.

For the purpose of this section we take a real network G , find parameters $\hat{\Theta}$ using KRONFIT, generate a synthetic graph K using $\hat{\Theta}$, and then compare G and K by comparing their properties that we introduced in section 5.2. In all experiments we started from a random point (random initiator matrix) and run gradient

descent for 100 steps. At each step we estimate the likelihood and the gradient based on 510,000 sampled permutations where we discard first 10,000 samples to allow the chain to burn-in.

Fitting to Autonomous Systems network

First, we focus on the Autonomous Systems network obtained from the University of Oregon Route Views project [RouteViews, 1997]. Given the AS network G we run KRONFIT to obtain parameter estimates $\hat{\Theta}$. Using the $\hat{\Theta}$ we then generate a synthetic Kronecker graph K , and compare the properties of G and K .

Figure 5.17 shows properties of AS-ROUTEVIEWS, and compares them with the properties of a synthetic Kronecker graph generated using the fitted parameters $\hat{\Theta}$ of size 2×2 . Notice that properties of both graphs match really well. The estimated parameters are $\hat{\Theta} = [0.987, 0.571; 0.571, 0.049]$.

Figure 5.17(a) compares the degree distributions of the AS-ROUTEVIEWS network and its synthetic Kronecker estimate. In this and all other plots we use the exponential binning which is a standard procedure the de-noise the data when plotting on log-log scales. Notice a very close match in degree distribution between the real graph and its synthetic counterpart.

Figure 5.17(b) plots the cumulative number of pairs of nodes $g(h)$ that can be reached in $\leq h$ hops. The hop plot gives a sense about the distribution of the shortest path lengths in the network and about the network diameter. Last, Figures 5.17(c) and (d) plot the spectral properties of the graph adjacency matrix. Figure 5.17(c) plots largest singular values vs. rank, and (d) plots the components of left singular vector (the network value) vs. the rank. Again notice good agreement with the real graph while using only four parameters.

Moreover, on all plots the error bars of two standard deviations show the variance of the graph properties for different realizations $R(\hat{\Theta}^{[k]})$. To obtain the error bars we took the same $\hat{\Theta}$, and generated 50 realizations of a Kronecker graph. As for the most of the plots the error bars are so small to be practically invisible, this shows that the variance of network properties when generating a Stochastic Kronecker graph is indeed very small.

Also notice that the AS-ROUTEVIEWS is an undirected graph, and that the fitted parameter matrix $\hat{\Theta}$ is in fact symmetric. This means that without a priori biasing the fitting towards undirected graphs, the recovered parameters obey this aspect of the network. Fitting AS-ROUTEVIEWS graph from a random set of parameters, performing gradient descent for 100 iterations and at each iteration sampling half a million permutations, took less than 10 minutes on a standard desktop PC. This is a significant speedup over [Bezáková et al., 2006], where by using a similar permutation sampling approach for calculating the likelihood of a preferential attachment model on similar AS-ROUTEVIEWS graph took about two days on a cluster of 50 machines, while in our case the computation took 10 minutes on a desktop PC.

Choice of the initiator matrix size N_1

As mentioned earlier for finding the optimal number of parameters, *i.e.*, selecting the size of initiator matrix, BIC criterion naturally applies to the case of Kronecker Graphs. Figure 5.23(b) shows BIC scores for the following experiment: We generated Kronecker graph with $N = 2,187$ and $E = 8,736$ using $N_1 = 3$ (9 parameters) and $k = 7$. For $1 \leq N_1 \leq 9$ we find the MLE parameters using gradient descent, and calculate the BIC scores. Model with the lowest score is chosen. As figure 5.23(b) shows we

N_1	$l(\hat{\Theta})$	N_1^k	E_1^k	$ \{\deg(u) > 0\} $	BIC score
2	-152,499	8,192	25,023	5,675	152,506
3	-127,066	6,561	28,790	5,683	127,083
4	-153,260	16,384	24,925	8,222	153,290
5	-149,949	15,625	29,111	9,822	149,996
6	-128,241	7,776	26,557	6,623	128,309
As-ROUTEVIEWS		26,467	6,474		

Table 5.2: Log-likelihood at MLE for different choices of the size of the initiator matrix N_1 for the As-ROUTEVIEWS graph. Notice the log-likelihood $l(\hat{\theta})$ generally increases with the model complexity N_1 . Also notice the effect of zero-padding, *i.e.*, for $N_1 = 4$ and $N_1 = 5$ the constraint of the number of nodes being an integer power of N_1 decreases the log-likelihood. However, the column $|\{\deg(u) > 0\}|$ gives the number of non-isolated nodes in the network which is much less than N_1^k and is in fact very close to the true number of nodes in the As-ROUTEVIEWS. Using the BIC scores we see that $N_1 = 3$ or $N_1 = 6$ are best choices for the size of the initiator matrix.

recovered the true model, *i.e.*, BIC score is the lowest for the model with the true number of parameters, $N_1 = 3$.

Intuitively we expect a more complex model with more parameters to fit the data better. Thus we expect larger N_1 to generally give better likelihood. On the other hand the fit will also depend on the size of the graph G . Kronecker graphs can only generate graphs on N_1^k nodes, while real graphs do not necessarily have N_1^k nodes (for some, preferably small, integers N_1 and k). To solve this problem we choose k so that $N_1^{k-1} < N(G) \leq N_1^k$, and then augment G by adding $N_1^k - N$ isolated nodes. Or equivalently, we pad the adjacency matrix of G with zeros until it is of the appropriate size, $N_1^k \times N_1^k$. While this solves the problem of requiring the integer power of the number of nodes it also makes the fitting problem harder as when $N \ll N_1^k$ we are basically fitting G plus a large number of isolated nodes.

Table 5.2 shows the results of fitting Kronecker graphs to AS-ROUTEVIEWS while varying the size of the initiator matrix N_1 . First, notice that in general larger N_1 results in higher log-likelihood $l(\hat{\Theta})$ at MLE. Similarly, notice (column N_1^k) that while AS-ROUTEVIEWS has 6,474 nodes, Kronecker estimates have up to 16,384 nodes ($16,384 = 4^7$, which is the first integer power of 4 greater than 6,474). However, we also show the number of non-zero degree (non-isolated) nodes in the Kronecker graph (column $|\{\deg(u) > 0\}|$). Notice that the number of non-isolated nodes well corresponds to the number of nodes in AS-ROUTEVIEWS network. This shows that KRONFIT is actually fitting the graph well, it successfully fits the structure of the graph plus a number of isolated nodes. Last, column E_1^k gives the number of edges in the corresponding Kronecker graph which is close to the true number of edges of the AS-ROUTEVIEWS graph.

Last, comparing the log-likelihood at MLE and the BIC score in Table 5.2 we notice that the log-likelihood heavily dominates the BIC score. This means that the size of the initiator matrix (number of parameters) is so small that one does not really have to care about overfitting. Thus we can just choose the initiator matrix that maximizes the likelihood. A simple calculation shows that one would need to take initiator matrices with thousands of entries before the model complexity part of BIC score would start to play a significant role.

We further examine the sensitivity of the choice of the initiator size by the following experiment. We

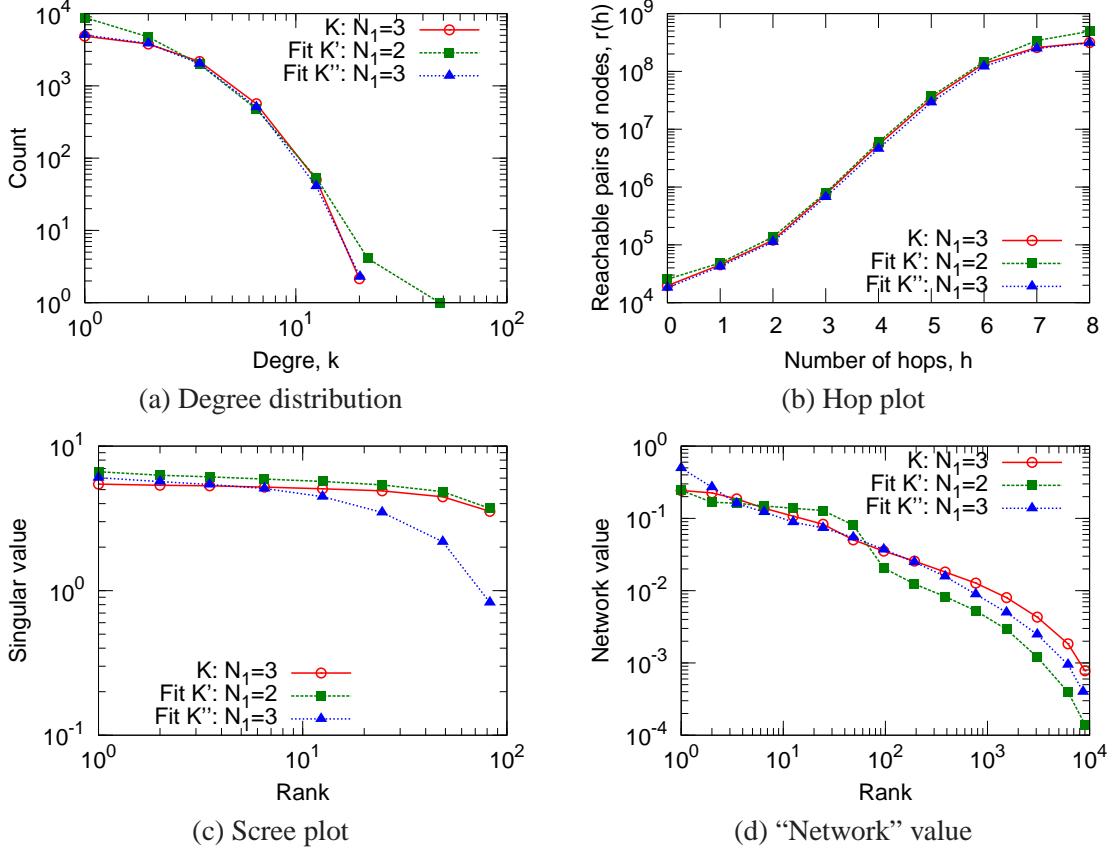


Figure 5.18: 3 by 3 Stochastic Kronecker Graph: Given a Stochastic Kronecker Graph G generated from $N_1 = 3$ (red curve), we fit a Kronecker graph K' with $N'_1 = 2$ (green) and K'' with $N''_1 = 3$ (blue). Not surprisingly K'' fits the properties of K perfectly as the model is of same complexity. On the other hand K' has only 4 parameters (instead of 9 as in K and K'') and still fits well.

generate a Stochastic Kronecker Graph K on 9 parameters ($N_1 = 3$), and then fit a Kronecker graph K' with a smaller number of parameters (4 instead of 9, $N'_1 = 2$). And also a Kronecker graph K'' of the same complexity as K ($N''_1 = 3$).

Figure 5.18 plots the properties of all three graphs. Not surprisingly K'' (blue) fits the properties of K (red) perfectly as the initiator is of the same size. On the other hand K' (green) is a simpler model with only 4 parameters (instead of 9 as in K and K'') and still generally fits well: hop plot and degree distribution match well, while spectral properties of graph adjacency matrix, especially scree plot, are not matched that well. This shows that nothing drastic happens and that even a bit too simple model still fits the data well. In general we observe empirically that by increasing the size of initiator matrix one does not gain radically better fits for degree distribution and hop plot. On the other hand there is usually an improvement in the scree plot and the plot of network values when one increases the initiator size.

Snapshot at time	N	E	$l(\hat{\Theta})$	Estimates at MLE, $\hat{\Theta}$
T_1	2,048	8,794	-40,535	[0.981, 0.633; 0.633, 0.048]
T_2	4,088	15,711	-82,675	[0.934, 0.623; 0.622, 0.044]
T_3	6,474	26,467	-152,499	[0.987, 0.571; 0.571, 0.049]

Table 5.3: Parameter estimates of the three temporal snapshots of the AS-ROUTEVIEWS network. Notice that estimates stay remarkably stable over time.

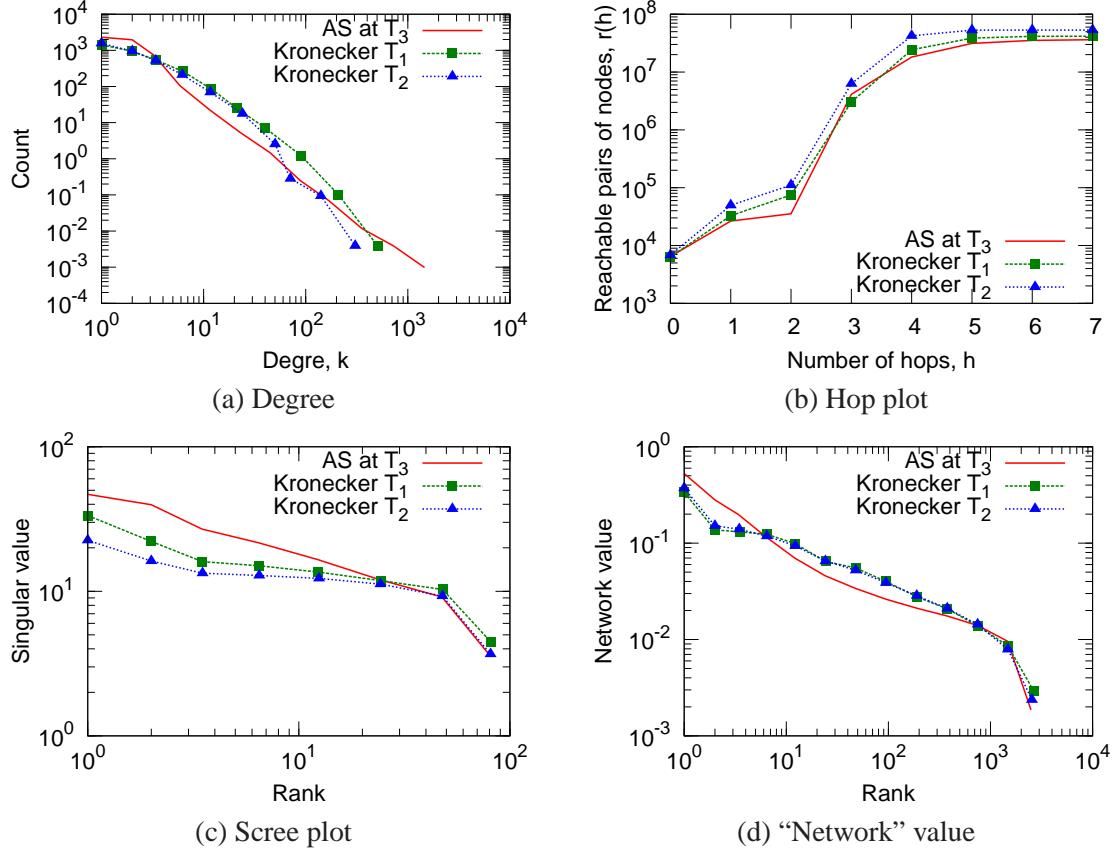


Figure 5.19: Autonomous systems network over time (AS-ROUTEVIEWS): Overlayed patterns of real AS-ROUTEVIEWS network at time T_3 and the Kronecker graphs with parameters estimated from AS-ROUTEVIEWS at time T_1 and T_2 . Notice good fits which means that parameters estimated on historic snapshots can be used to estimate the graph in the future.

Network parameters over time

Next we briefly examine the evolution of the Kronecker initiator for a temporally evolving graph. The idea is that given parameter estimates of a real-graph G_t at time t , we can forecast the future structure of the graph G_{t+x} at time $t+x$, i.e., using parameters obtained from G_t we can generate a larger synthetic graph K that will be similar to G_{t+x} .

As we have the information about the evolution of the AS-ROUTEVIEWS network, we estimated parameters for three snapshots of the network when it had about 2^k nodes. Table 5.3 gives the results of the fitting for the three temporal snapshots of the AS-ROUTEVIEWS network. Notice the parameter estimates $\hat{\Theta}$ remain remarkably stable over time. This means that Kronecker graphs can be used to estimate the

Network	N	E	Estimated parameters $\hat{\Theta}$	$l(\hat{\Theta})$	Time
AS-ROUTEVIEWS	6,474	26,467	[0.987, 0.571; 0.571, 0.049]	-152, 499	8m15s
ATP-GR-QC	19,177	26,169	[0.902, 0.253; 0.221, 0.582]	-242, 493	7m40s
BIO-PROTEINS	4,626	29,602	[0.847, 0.641; 0.641, 0.072]	-185, 130	43m41s
EMAIL-INSIDE	986	32,128	[0.999, 0.772; 0.772, 0.257]	-107, 283	1h07m
CA-GR-QC	5,242	28,980	[0.999, 0.245; 0.245, 0.691]	-160, 902	14m02s
AS-NEWMAN	22,963	96,872	[0.954, 0.594; 0.594, 0.019]	-593, 747	28m48s
BLOG-NAT05-6M	31,600	271,377	[0.999, 0.569; 0.502, 0.221]	-1, 994, 943	47m20s
BLOG-NAT06ALL	32,443	318,815	[0.999, 0.578; 0.517, 0.221]	-2, 289, 009	52m31s
CA-HEP-PH	12,008	237,010	[0.999, 0.437; 0.437, 0.484]	-1, 272, 629	1h22m
CA-HEP-TH	9,877	51,971	[0.999, 0.271; 0.271, 0.587]	-343, 614	21m17s
CIT-HEP-PH	30,567	348,721	[0.994, 0.439; 0.355, 0.526]	-2, 607, 159	51m26s
CIT-HEP-TH	27,770	352,807	[0.990, 0.440; 0.347, 0.538]	-2, 507, 167	15m23s
EPINIONS	75,879	508,837	[0.999, 0.532; 0.480, 0.129]	-3, 817, 121	45m39s
GNUTELLA-25	22,687	54,705	[0.746, 0.496; 0.654, 0.183]	-530, 199	16m22s
GNUTELLA-30	36,682	88,328	[0.753, 0.489; 0.632, 0.178]	-919, 235	14m20s
DELICIOUS	205,282	436,735	[0.999, 0.327; 0.348, 0.391]	-4, 579, 001	27m51s
ANSWERS	598,314	1,834,200	[0.994, 0.384; 0.414, 0.249]	-20, 508, 982	2h35m
CA-DBLP	425,957	2,696,489	[0.999, 0.307; 0.307, 0.574]	-26, 813, 878	3h01m
FLICKR	584,207	3,555,115	[0.999, 0.474; 0.485, 0.144]	-32, 043, 787	4h26m
WEB-NOTREDAME	325,729	1,497,134	[0.999, 0.414; 0.453, 0.229]	-14, 588, 217	02h59m

Table 5.4: Results of parameter estimation for 20 different networks. Tables in Section A.2 give the description and basic properties of the above network datasets.

structure of the networks in the future, *i.e.*, parameters estimated from the historic data can extrapolate the graph structure in the future.

Figure 5.19 further explores this. It overlays the graph properties of the real AS-ROUTEVIEWS network at time T_3 and the synthetic graphs for which we used the parameters obtained on historic snapshots of AS-ROUTEVIEWS at times T_1 and T_2 . The agreements are good which demonstrates that Kronecker graphs can forecast the structure of the network in the future.

Moreover, this experiments also shows that parameter estimates do not suffer much from the zero padding of graph adjacency matrix (*i.e.*, adding isolated nodes to make G have N_1^k nodes). Snapshots of AS-ROUTEVIEWS at T_1 and T_2 have close to 2^k nodes, while we had to add 26% (1,718) isolated nodes to the network at T_3 to make the number of nodes be 2^k . Regardless of this we see the parameter estimates $\hat{\Theta}$ remain basically constant over time, which seems to be independent of the number of isolated nodes added. This means that the estimated parameters are not biased too much from zero padding the adjacency matrix of G .

5.6.5 Fitting to other large real-world networks

Last, we present results of fitting Stochastic Kronecker Graph to 20 large real-world networks: large online social networks, like EPINIONS, FLICKR and DELICIOUS, web and blog graphs (WEB-NOTREDAME, BLOG-NAT05-6M, BLOG-NAT06ALL), internet and peer-to-peer networks (AS-NEWMAN, GNUTELLA-25, GNUTELLA-30), collaboration networks of co-authorships from DBLP (CA-DBLP) and various areas of physics (CA-HEP-TH, CA-HEP-PH, CA-GR-QC), physics citation networks (CIT-HEP-PH, CIT-

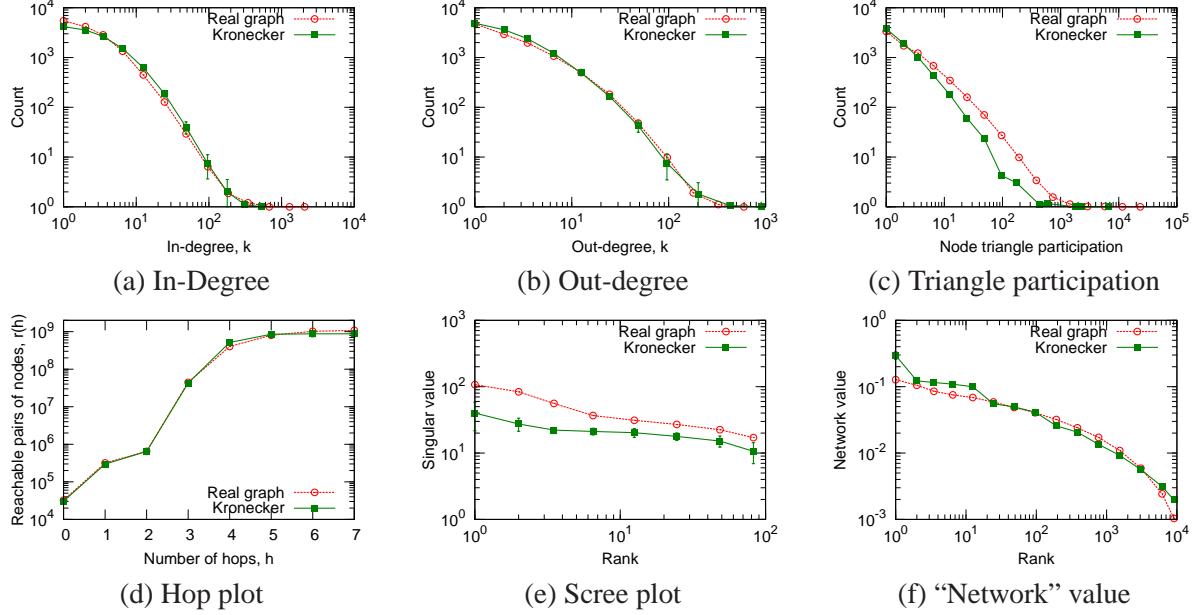


Figure 5.20: *Blog network (BLOG-NAT06ALL):* Overlayed patterns of real network and the estimated Kronecker graph using 4 parameters (2×2 initiator matrix). Notice that the Kronecker graph matches all properties of the real network.

HEP-TH), an email network (EMAIL-INSIDE), a protein interaction network BIO-PROTEINS, and a bipartite affiliation network (authors-to-papers, ATP-GR-QC). Refer to table A.2 in the appendix for the description and basic properties of these networks.

For each dataset we started gradient descent from a random point (random initiator matrix) and run it for 100 steps. At each step we estimate the likelihood and the gradient based on 510,000 sampled permutations where we discard first 10,000 samples to allow the chain to burn-in.

Table 5.4 gives the estimated parameters, the corresponding log-likelihoods and the wall clock times. All experiments were carried out on standard desktop computer. Notice that the estimated initiator matrix $\hat{\Theta}$ seems to have almost universal structure with a big value in the top left entry, a very low value at the bottom right corner and intermediate values in the other two corners. We further discuss the implications of such structure of Kronecker initiator matrix on the global network structure in the next section.

Last, Figures 5.20 and 5.21 show overlays of various network properties of real and the estimated synthetic networks. In addition to the network properties we plotted in Figure 5.18, we also separately plot in- and out-degree distributions (as both networks are directed) and plot the node triangle participation in panel (c), where we plot the number of triangles a node participates in versus the number of such nodes. (Again the error bars show the variance of network properties over different realizations $R(\hat{\Theta}^{[k]})$ of a Stochastic Kronecker graph.)

Notice that for both networks and in all cases the properties of the real network and the synthetic Kronecker coincide really well. Using Stochastic Kronecker Graph with just 4 parameters we match the scree plot, degree distributions, triangle participation, hop plot and network values.

Given the experience from the Autonomous systems we only present the results for the simplest model

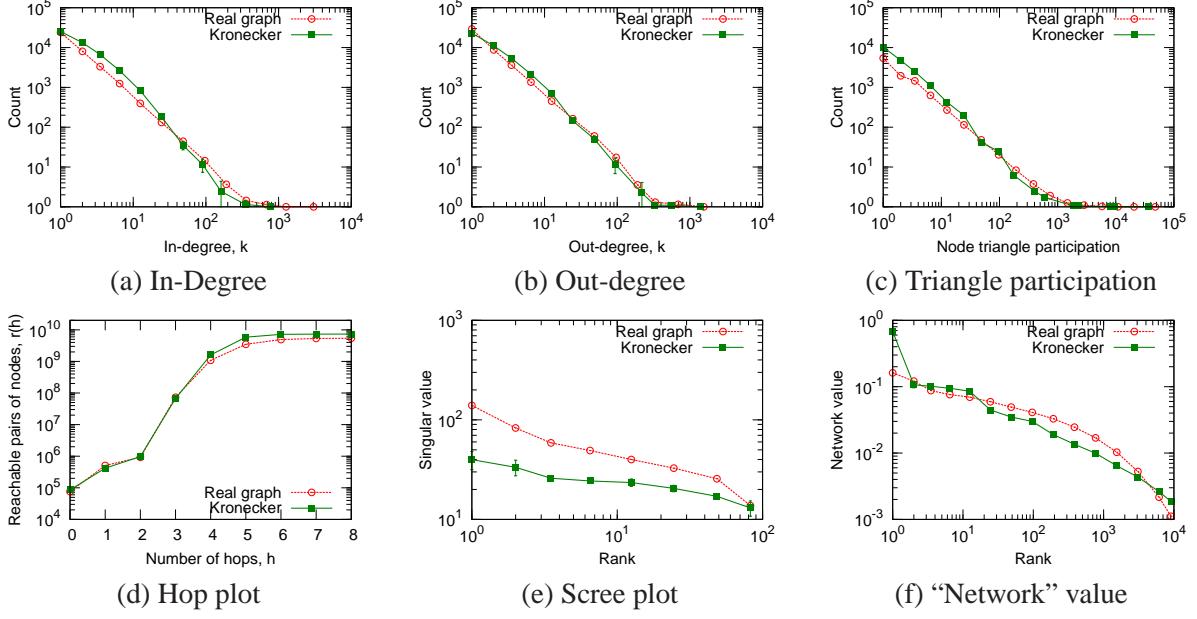


Figure 5.21: EPINIONS *who-trusts-whom* social network: Overlayed patterns of real network and the fitted Kronecker graph using only 4 parameters (2×2 initiator matrix). Again, the synthetic Kronecker graph matches all the properties of the real network.

with initiator size $N_1 = 2$. Empirically we also observe that $N_1 = 2$ gives surprisingly good fits and the estimation procedure is the most robust and converges the fastest. Using larger initiator matrices $N_1 > 2$ generally helps improve the likelihood but not dramatically. In terms of matching the network properties we also get a slight improvement by making the model more complex. Figure 5.22 gives the percent improvement in log-likelihood as we make the model more complex. We use the log-likelihood of a 2×2 model as a baseline and estimate the log-likelihood at MLE for larger initiator matrices. Again, models with more parameters tend to fit better. However, sometimes due to zero-padding of graph adjacency matrix they actually have lower log-likelihood.

5.6.6 Scalability

Last we also empirically evaluate the scalability of the KRONFIT. The experiment confirms that KRONFIT runtime scales linearly with the number of edges E in a graph G . More precisely, we performed the following experiment.

We generated a sequence of increasingly larger synthetic graphs on N nodes and $8N$ edges, and measured the time of one iteration of gradient descent, *i.e.*, sample 1 million permutations and evaluate the gradients. We started with a graph on 1,000 nodes, and finished with a graph on 8 million nodes, and 64 million edges. Figure 5.23(a) shows KRONFIT scales *linearly* with the size of the network. We plot wall-clock time vs. size of the graph. Dashed line presents linear fit to the data points.

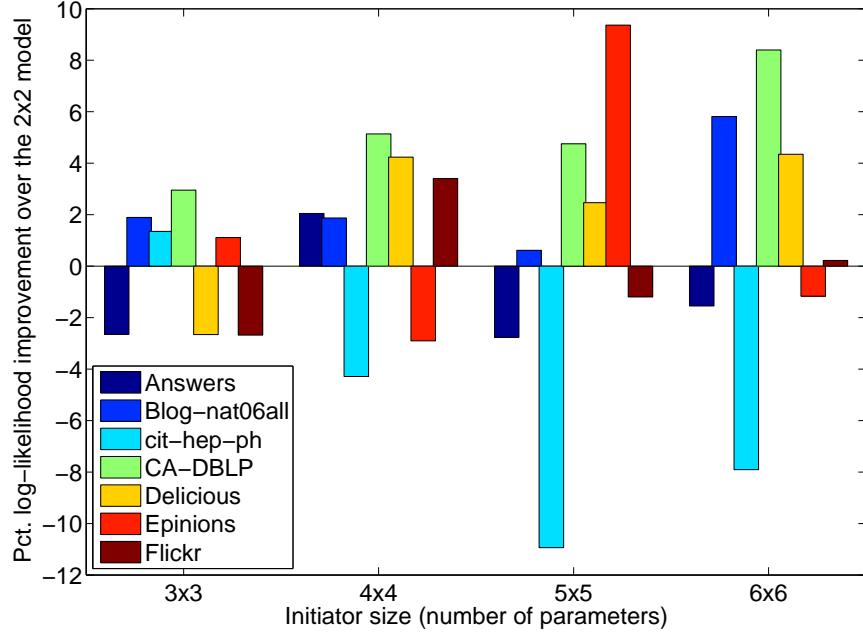


Figure 5.22: Percent improvement in log-likelihood over the 2×2 model as we increase the model complexity (size of initiator matrix). In general larger initiator matrices that have more degrees of freedom help improving the fit of the model.

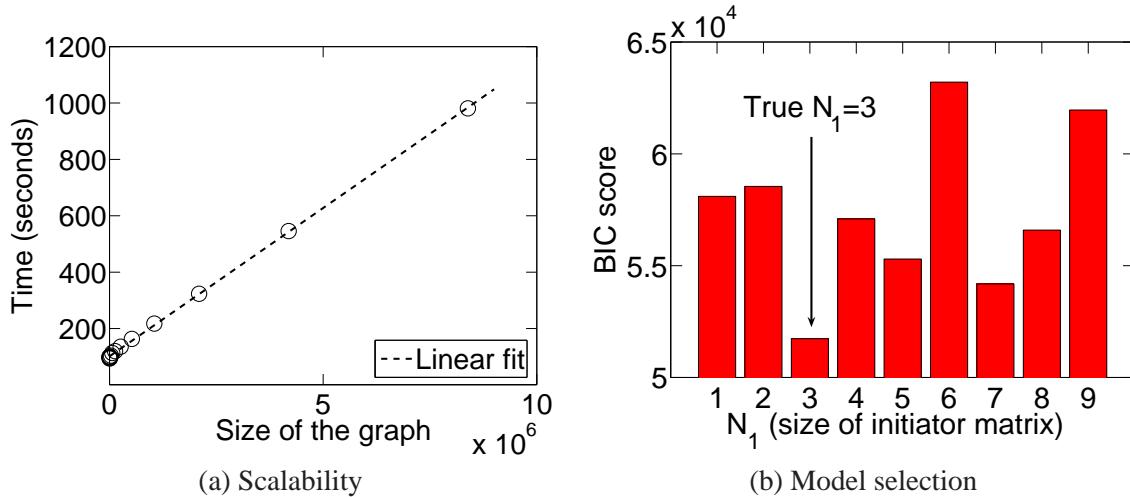


Figure 5.23: (a) Processor time to sample 1 million gradients as the graph grows. Notice the algorithm scales linearly with the graph size. (b) BIC score for model selection.

5.7 Discussion

Here we discuss several of the desirable properties of the proposed Kronecker Graphs.

Generality: Stochastic Kronecker Graphs include several other generators as special cases: For $\theta_{ij} = c$, we obtain classical Erdős-Rényi random graph model; for $\theta_{i,j} \in \{0, 1\}$, we obtain a deterministic Kro-

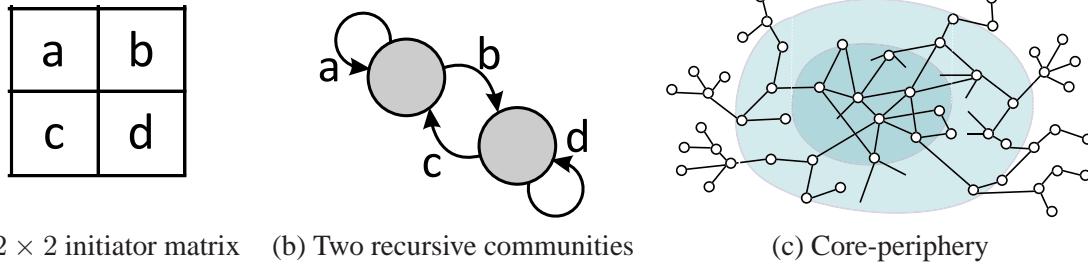


Figure 5.24: 2×2 Kronecker initiator matrix (a) can be thought of as two communities where there are a and d edges inside each of the communities and b and c edges crossing the communities as illustrated in (b). The each sub-community can then be recursively divided using the same pattern. (c) The onion like core-periphery structure where the network gets denser and denser as we move towards the center of the network.

necker graph; setting the K_1 matrix to a 2×2 matrix, we obtain the RMAT generator [Chakrabarti et al., 2004]. In contrast to Kronecker graphs, the RMAT cannot extrapolate into the future, since it needs to know the number of edges to insert. Thus, it is incapable of obeying the densification power law.

Phase transition phenomena: The Erdős-Rényi graphs exhibit phase transitions [Erdős and Rényi, 1960]. Several researchers argue that real systems are “at the edge of chaos” [Bak, 1996, Sole and Goodwin, 2000]. Stochastic Kronecker Graphs also exhibit phase transitions [Mahdian and Xu, 2007] for the emergence of the giant component and another phase transition for connectivity.

Implications to the structure of the large-real networks: Empirically we found that 2×2 initiator ($N_1 = 2$) fits well the properties of real-world networks. Moreover, given a 2×2 initiator matrix, one can look at it as a recursive expansion of two groups into sub-groups. We introduced this recursive view of Kronecker graphs back in section 5.3. So, one can then interpret the diagonal values of Θ as the proportion of edges inside each of the groups, and the off-diagonal values give the fraction of edges connecting the groups. Figure 5.24 illustrates the setting for two groups.

For example, as shown in Figure 5.24, large a, d and small b, c would imply that the network is composed of hierarchically nested communities, where there are many edges inside each community and few edges crossing them. One could think of this structure as some kind of organizational or university hierarchy, where one expects the most friendships between people within same lab, a bit less between people in the same department, less across different departments, and the least friendships to be formed across people from different schools of the university.

However, parameter estimates for a wide range of networks presented in Table 5.4 suggests a very different picture of the network structure. Notice that for most networks $a \gg b > c \gg d$. Moreover, $a \approx 1$, $b \approx c \approx 0.6$ and $d \approx 0.2$. We empirically observed that the same structure of initiator matrix $\hat{\Theta}$ also holds when fitting 3×3 or 4×4 models. Always the top left element is the largest and then the values on the diagonal decay faster than off the diagonal.

This suggests a network structure which is also known as *core-periphery* [Borgatti and Everett, 2000, Holme, 2005], the *jellyfish* [Tauro et al., 2001, Siganos et al., 2006], or the *octopus* [Chung and Lu, 2006a] structure of the network as illustrated in Figure 5.24(c).

All of the above basically say that the network is composed of a densely linked network core and the

periphery. In our case this would imply the following structure of the initiator matrix. Core is modeled by parameter a and the periphery by d . The most edges are inside the core (large a), and the fewest between the nodes of periphery (small d). Then there are many more edges between the core and the periphery than inside the periphery ($b, c > d$). This is exactly what we see. Many edges are inside the core (large a), there are very few edges among the periphery nodes (small d), while there are relatively many edges connecting the core with the periphery (b, c are relatively large). And in spirit of Kronecker graphs the structure repeats recursively — core has again the dense core and the periphery, and so on. And similarly the periphery itself has the core and the periphery.

This suggest an “onion” like network structure as illustrated in Figure 5.24(c), where the network is composed of denser and denser layers as one moves towards the center of the network. We also observe similar structure of the Kronecker initiator when fitting 3×3 or 4×4 initiator matrix. The diagonal elements have large but decreasing values with off diagonal elements following same decreasing pattern.

One of the implications of this is that networks do not break nicely into hierarchically organized sets of communities that nicely allow themselves to partitioning and community identification algorithms. On contrary, this suggests that large networks can be decomposed into a densely linked core with many small periphery pieces hanging off the core. This is in accordance with our recent results [Leskovec et al., 2008b], that make similar observation (but based on a completely different methodology) about the structure of large real-world networks. We further explore this in greater detail in chapter 10.

5.8 Conclusion

In conclusion, the main contribution of this work is a family of models of network structure that uses a non-traditional matrix operation, the *Kronecker product*. The resulting graphs (a) have all the static properties (heavy-tailed degree distribution, small diameter, etc.), (b) all the temporal properties (densification, shrinking diameter) that are found in real networks. And in addition, (c) we can formally prove all of these properties.

Several of the proofs are extremely simple, thanks to the rich theory of Kronecker multiplication. We also provide proofs about the diameter and effective diameter, and we show that Stochastic Kronecker Graphs can mimic real graphs well.

Moreover, we also presented KRONFIT, a fast, scalable algorithm to estimate Stochastic Kronecker initiator, which can be then used to create a synthetic graph that mimics the properties of a given real network.

In contrast to earlier work, our work has the following novelties: (a) it is among the few that estimates the parameters of the chosen generator in a principled way, (b) it is among the few that has a concrete measure of goodness of the fit (namely, the likelihood), (c) it avoids the quadratic complexity of computing the likelihood by exploiting the properties of the Kronecker graphs, and (d) it avoids the factorial explosion of the node correspondence problem, by using the Metropolis sampling.

The resulting algorithm matches well all the known properties of real graphs, as we show with the Epinions graph and the AS graph, it scales linearly on the number of edges, and it is orders of magnitudes faster than earlier graph-fitting attempts: 20 minutes on a commodity PC, versus 2 days on a cluster of 50 workstations [Bezáková et al., 2006].

The benefits of fitting a Kronecker graph model into a real graph are several:

- *Extrapolation*: Once we have the Kronecker generator Θ for a given real matrix G (such that G is mimicked by $\Theta^{[k]}$), a larger version of G can be generated by $\Theta^{[k+1]}$.
- *Null-model*: When analyzing a real network G one often needs to assess the significance of the observation. $\Theta^{[k]}$ that mimics G can be used as an accurate model of G .
- *Network structure*: fitted parameters give insight into the global network and community structure of the network.
- *Forecasting*: As we demonstrated one can obtain Θ from a graph G_t at time t such that G is mimicked by $\Theta^{[k]}$. Then Θ can be used to model the structure of G_{t+x} in the future.
- *Sampling*: Similarly, if we want a realistic sample of the real graph, we could use a smaller exponent in the Kronecker exponentiation, like $\Theta^{[k-1]}$.
- *Anonymization*: Since $\Theta^{[k]}$ mimics G , we can publish $\Theta^{[k]}$, without revealing information about the nodes of the real graph G .

Part 1 – Network evolution: Conclusion

Despite the enormous recent interest in large-scale network data, and the range of interesting patterns identified for static snapshots of graphs (*e.g.*, heavy-tailed distributions, small-world phenomena), there has been relatively little work on the properties of the time evolution of real graphs. This was exactly the focus of this part of the thesis.

Observations: In contrast to the standard modeling assumption that the average out-degree remains constant over time, we discovered that real graphs have out-degrees that grow over time, following a *Densification power law*. Moreover, our experiments also show that the standard assumption of slowly growing diameters does not hold in a range of real networks; rather, the *diameter* may actually exhibit a *gradual decrease* as the network grows. We then developed the Forest Fire Model, based on only two parameters, where the observed patterns naturally *emerge* from simple local rules that govern individual edge creation.

Models: We then presented a detailed study of network evolution by analyzing four large online social networks with full temporal information about individual node and edge arrivals. The use of the *maximum-likelihood* principle allowed us to quantify the bias of new edges towards the degree and age of nodes, and to objectively compare various models such as preferential attachment. In fact, our work is the first to directly quantify the amount of preferential attachment that occurs in large social networks. Based on our observations, we derived an extremely simple yet surprisingly accurate model of network evolution, that *fully specifies three essential processes* taking place in evolving networks: (a) node arrivals, (b) edges arrivals, and (c) edge placement.

Algorithms: Last, we presented a family of models of network structure that uses a non-traditional matrix operation, the *Kronecker product*. We show that resulting graphs (a) have all the static properties (heavy-tailed degree distribution, small diameter), (b) all the temporal properties (densification, shrinking diameter), and in addition, (c) we can formally prove all of these properties. Moreover, we also presented KRONFIT, a fast, scalable algorithm to estimate Kronecker initiator, which can be then used to create a synthetic graph that mimics the properties of a given real graph. Naive approach to fitting would take super-exponential time, while KRONFIT takes *linear* time, by exploiting the structure of Kronecker matrix multiplication and by using sampling. In contrast to earlier work, Kronecker graphs are *mathematically tractable* model of network generation satisfying many real network properties, while we can also efficiently fit it to graphs on millions of nodes and edges.

Part II

Network cascades

**How do influence and information spread over
the network, and
how to detect this quickly?**

Part 2 – Network cascades: Overview

A basic premise behind the study of social networks is that interaction leads to complex collective behavior. Cascades are a form of collective behavior that has been analyzed both empirically and theoretically, but for which the study of complete, large-scale datasets has been limited. Here we show that cascades exist in a large real-world networks, and investigate some of their structural features.

We present two studies of diffusion and cascading behavior in networks, where for the first time we are able to directly measure millions of propagations individually.

Observations: First, we study the influence and recommendation propagation in a large viral marketing network. And then present our work on the information propagation on the web and the cascades this process results in. We make observations on the sizes, *shapes*, and temporal characteristics of the cascades. We also explore what product and recommendation network factors play a role in the propagation and purchases of products, and notice that the human adoption curve follows *diminishing returns* trend, as opposed to the critical threshold conjecture.

Models: We also analyzed one of the largest available collections of blog information. We investigate how blogs behave and how *information propagates* through the blogosphere. We develop a simple but accurate model of information propagation on the blogosphere. In contrast with viral marketing stars and chains are basic components of blog cascades, with stars being more common.

Algorithms: As we observe the cascades spreading through the network a natural question is how to detect them effectively. For example, given a water distribution network, where should we place sensors to quickly detect contaminants? Or, which blogs should we read to avoid missing important stories? These seemingly different problems share common structure: *Outbreak detection* can be modeled as selecting nodes (sensor locations, blogs) in a network, in order to detect the spreading of a virus or information as quickly as possible. We present a general methodology for *near optimal* sensor placement in these and related problems. We demonstrate that many realistic outbreak detection objectives exhibit the property of “submodularity”. We exploit submodularity to develop an efficient algorithm that scales to large problems, achieving near optimal placements, while being *700 times* faster than a simple greedy algorithm. We also derive online bounds on the quality of the placements obtained by *any* algorithm. Our algorithms and bounds also handle cases where nodes (sensor locations, blogs) have different costs.

Chapter 6

Diffusion and cascading behavior in viral marketing

6.1 Introduction

With consumers showing increasing resistance to traditional forms of advertising such as TV or newspaper ads, marketers have turned to alternate strategies, including viral marketing. Viral marketing exploits existing social networks by encouraging customers to share product information with their friends. Previously, a few in depth studies have shown that social networks affect the adoption of individual innovations and products (for a review see [Rogers, 1995] or [Strang and Soule, 1998]). But until recently it has been difficult to measure how influential person-to-person recommendations actually are over a wide range of products. Moreover, Subramani and Rajagopalan [Subramani and Rajagopalan, 2003] noted that “there needs to be a greater understanding of the contexts in which viral marketing strategy works and the characteristics of products and services for which it is most effective. This is particularly important because the inappropriate use of viral marketing can be counterproductive by creating unfavorable attitudes towards products. What is missing is an analysis of viral marketing that highlights systematic patterns in the nature of knowledge-sharing and persuasion by influencers and responses by recipients in online social networks.”

Here we were able to in detail study the above mentioned problem. We were able to directly measure and model the effectiveness of recommendations by studying one online retailer’s incentivised viral marketing program. The website gave discounts to customers recommending any of its products to others, and then tracked the resulting purchases and additional recommendations.

Although word of mouth can be a powerful factor influencing purchasing decisions, it can be tricky for advertisers to tap into. Some services used by individuals to communicate are natural candidates for viral marketing, because the product can be observed or advertised as part of the communication. Email services such as Hotmail and Yahoo had very fast adoption curves because every email sent through them contained an advertisement for the service and because they were free. Hotmail spent a mere \$50,000 on traditional marketing and still grew from zero to 12 million users in 18 months [Jurvetson, 2000]. The Hotmail user base grew faster than any media company in history – faster than CNN, faster than AOL, even faster than Seinfeld’s audience. By mid-2000, Hotmail had over 66 million users with 270,000 new accounts being established each day [Bronson, 1998]. Google’s Gmail also captured a significant part of market share in spite of the fact that the *only* way to sign up for the service was through a referral.

Most products cannot be advertised in such a direct way. At the same time the choice of products available to consumers has increased manyfold thanks to online retailers who can supply a much wider variety of products than traditional brick-and-mortar stores. Not only is the variety of products larger, but one observes a ‘fat tail’ phenomenon, where a large fraction of purchases are of relatively obscure items. On Amazon.com, somewhere between 20 to 40 percent of unit sales fall outside of its top 100,000 ranked products [Brynjolfsson et al., 2003]. Rhapsody, a streaming-music service, streams more tracks outside than inside its top 10,000 tunes [Anonymous, 2005]. Some argue that the presence of the long tail indicates that niche products with low sales are contributing significantly to overall sales online.

We find that product purchases that result from recommendations are not far from the usual 80-20 rule. The rule states that the top twenty percent of the products account for 80 percent of the sales. In our case the top 20% of the products contribute to about half the sales.

Effectively advertising these niche products using traditional advertising approaches is impractical. Therefore using more targeted marketing approaches is advantageous both to the merchant and the consumer, who would benefit from learning about new products.

The problem is partly addressed by the advent of online product and merchant reviews, both at retail sites such as EBay and Amazon, and specialized product comparison sites such as Epinions and CNET. Of further help to the consumer are collaborative filtering recommendations of the form “people who bought x also bought y ” feature [Linden et al., 2003]. These refinements help consumers discover new products and receive more accurate evaluations, but they cannot completely substitute personalized recommendations that one receives from a friend or relative. It is human nature to be more interested in what a friend buys than what an anonymous person buys, to be more likely to trust their opinion, and to be more influenced by their actions. As one would expect our friends are also acquainted with our needs and tastes, and can make appropriate recommendations. A Lucid Marketing survey found that 68% of individuals consulted friends and relatives before purchasing home electronics – more than the half who used search engines to find product information [Burke, 2003].

In our study we are able to directly observe the effectiveness of person to person word of mouth advertising for hundreds of thousands of products for the first time. We find that most recommendation chains do not grow very large, often terminating with the initial purchase of a product. However, occasionally a product will propagate through a very active recommendation network. We propose a simple stochastic model that seems to explain the propagation of recommendations.

Moreover, the characteristics of recommendation networks influence the purchase patterns of their members. For example, individuals’ likelihood of purchasing a product initially increases as they receive additional recommendations, but a saturation point is quickly reached. Interestingly, as more recommendations are sent between the same two individuals, the likelihood that they will be heeded decreases.

We find that communities (automatically found by a community finding algorithm) were usually centered around a product group, such as books, music, or DVDs, but almost all of them shared recommendations for all types of products. We also find patterns of homophily, the tendency of like to associate with like, with communities of customers recommending types of products reflecting their common interests.

We propose models to identify products for which viral marketing is effective: We find that the category and price of product plays a role, with recommendations of expensive products of interest to small, well connected communities resulting in a purchase more often. We also observe patterns in the timing of recommendations and purchases corresponding to times of day when people are likely to be shopping online or reading email.

We report on these and other findings in the following sections. We first survey the related work in section 6.2. We then describe the characteristics of the incentivised recommendations program and the dataset in section 6.3. Section 6.3.3 studies the temporal and static characteristics of the recommendation network. We investigate the propagation of recommendations and model the cascading behavior in section 6.4. Next we concentrate on the various aspects of the recommendation success from the viewpoint of the sender and the recipient of the recommendation in section 6.5. The timing and the time lag between the recommendations and purchases is studied in section 6.6. We study network communities, product characteristics and the purchasing behavior in section 6.7. Last, in section 6.8 we present a model that relates product characteristics and the surrounding recommendation network to predict the product recommendation success. We discuss the implications of our findings and conclude in section 6.10.

6.2 Connection to viral marketing

Viral marketing can be thought of as a diffusion of information about the product and its adoption over the network. Primarily in social sciences there is a long history of the research on the influence of social networks on innovation and product diffusion. However, such studies have been typically limited to small networks and typically a single product or service. For example, Brown and Reingen [Brown and Reingen, 1987] interviewed the families of students being instructed by three piano teachers, in order to find out the network of referrals. They found that strong ties, those between family or friends, were more likely to be activated for information flow and were also more influential than weak ties [Granovetter, 1973] between acquaintances. Similar observations were also made by DeBruyn and Lilien in [DeBruyn and Lilien, 2004] in the context of electronic referrals. They found that characteristics of the social tie influenced recipients' behavior but had different effects at different stages of decision making process: tie strength facilitates awareness, perceptual affinity triggers recipients' interest, and demographic similarity had a positive influence on each stage of the decision-making process.

Social networks can be composed by using various information, *i.e.*, geographic similarity, age, similar interests and so on. Yang and Allenby [Yang and Allenby, 2003] showed that the geographically defined network of consumers is more useful than the demographic network for explaining consumer behavior in purchasing Japanese cars. A recent study by Hill et al. [Hill et al., 2006] found that adding network information, specifically whether a potential customer was already “talking to” an existing customer, was predictive of the chances of adoption of a new phone service option. For the customers linked to a prior customer the adoption rate was 3–5 times greater than the baseline.

Factors that influence customers' willingness to actively share the information with others via word of mouth have also been studied. Frenzen and Nakamoto [Frenzen and Nakamoto, 1993] surveyed a group of people and found that the stronger the moral hazard presented by the information, the stronger the ties must be to foster information propagation. Also, the network structure and information characteristics interact when individuals form decisions about transmitting information. Bowman and Narayandas [Bowman and Narayandas, 2001] found that self-reported loyal customers were more likely to talk to others about the products when they were dissatisfied, but not more likely when they were satisfied.

In the context of the internet word-of-mouth advertising is not restricted to pairwise or small-group interactions between individuals. Rather, customers can share their experiences and opinions regarding a product with everyone. Quantitative marketing techniques have been proposed [Montgomery, 2001] to describe product information flow online, and the rating of products and merchants has been shown to effect the likelihood of an item being bought [Resnick and Zeckhauser, 2002, Chevalier and Mayzlin,

2006]. More sophisticated online recommendation systems allow users to rate others' reviews, or directly rate other reviewers to implicitly form a trusted reviewer network that may have very little overlap with a person's actual social circle. Richardson and Domingos [Domingos and Richardson, 2001, Richardson and Domingos, 2002b] used Epinions' trusted reviewer network to construct an algorithm to maximize viral marketing efficiency assuming that individuals' probability of purchasing a product depends on the opinions on the trusted peers in their network. Kempe, Kleinberg and Tardos [Kempe et al., 2003] have followed up on Richardson and Domingos' challenge of maximizing viral information spread by evaluating several algorithms given various models of adoption we discuss next.

Most of the previous research on the flow of information and influence through the networks has been done in the context of epidemiology and the spread of diseases over the network. See the works of Bailey [Bailey, 1975] and Anderson and May [Anderson and May, 2002] for reviews of this area. The classical disease propagation models are based on the stages of a disease in a host: a person is first *susceptible* to a disease, then if she is exposed to an infectious contact she can become *infected* and thus *infectious*. After the disease ceases the person is *recovered* or *removed*. The person is then *immune* for some period. The immunity can also wear off and the person becomes again susceptible. Thus SIR (susceptible – infected – recovered) models diseases where a recovered person never again becomes susceptible, while SIRS (SIS, susceptible – infected – (recovered) – susceptible) models population in which recovered host can become susceptible again. Given a network and a set of infected nodes the *epidemic threshold* is studied, *i.e.*, conditions under which the disease will either dominate or die out. In our case SIR model would correspond to the case where a set of initially infected nodes corresponds to people that purchased a product without first receiving the recommendations. A node can purchase a product only once, and then tries to infect its neighbors with a purchase by sending out the recommendations. SIS model corresponds to the less realistic case where a person can purchase a product multiple times as a result of multiple recommendations. The problem with these type of models is that they assume a known social network over which the diseases (product recommendations) are spreading and usually a single parameter which specifies the infectiousness of the disease. In our context this would mean that the whole population is equally susceptible to recommendations of a particular product.

There are numerous other models of influence spread in social networks. One of the first and most influential diffusion models was proposed by Bass [Bass, 1969]. The model of product diffusion predicts the number of people who will adopt an innovation over time. It does not explicitly account for the structure of the social network but it rather assumes that the rate of adoption is a function of the current proportion of the population who have already adopted (purchased a product in our case). The diffusion equation models the cumulative proportion of adopters in the population as a function of the intrinsic adoption rate, and a measure of social contagion. The model describes an S-shaped curve, where adoption is slow at first, takes off exponentially and flattens at the end. It can effectively model word-of-mouth product diffusion at the aggregate level, but not at the level of an individual person, which is one of the topics we explore in this chapter.

Diffusion models that try to model the process of adoption of an idea or a product can generally be divided into two groups:

- *Threshold model* [Granovetter, 1978] where each node in the network has a threshold $t \in [0, 1]$, typically drawn from some probability distribution. We also assign *connection weights* $w_{u,v}$ on the edges of the network. A node adopts the behavior if a sum of the connection weights of its neighbors that already adopted the behavior (purchased a product in our case) is greater than the threshold: $t \leq \sum_{\text{adopters}(u)} w_{u,v}$.

- *Cascade model* [Goldenberg et al., 2001] where whenever a neighbor v of node u adopts, then node u also adopts with probability $p_{u,v}$. In other words, every time a neighbor of u purchases a product, there is a chance that u will decide to purchase as well.

In the independent cascade model, Goldenberg et al. [Goldenberg et al., 2001] simulated the spread of information on an artificially generated network topology that consisted both of strong ties within groups of spatially proximate nodes and weak ties between the groups. They found that weak ties were important to the rate of information diffusion. Centola and Macy [Centola and Macy, 2005] modeled product adoption on small world topologies when a person’s chance of adoption is dependent on having more than one contact who had previously adopted. Wu and Huberman [Wu and Huberman, 2004b] modeled opinion formation on different network topologies, and found that if highly connected nodes were seeded with a particular opinion, this would proportionally effect the long term distribution of opinions in the network. Holme and Newman [Holme and Newman, 2006] introduced a model where individuals’ preferences are shaped by their social networks, but their choices of whom to include in their social network are also influenced by their preferences.

While these models address the question of how influence spreads in a network, they are based on *assumed* rather than *measured* influence effects. In contrast, our study tracks the actual diffusion of recommendations through email, allowing us to quantify the importance of factors such as the presence of highly connected individuals, or the effect of receiving recommendations from multiple contacts. Compared to previous empirical studies which tracked the adoption of a single innovation or product, our data encompasses over half a million different products, allowing us to model a product’s suitability for viral marketing in terms of both the properties of the network and the product itself.

6.3 The recommendation network

Here we briefly describe our viral marketing dataset and the properties of the recommendation network.

6.3.1 Recommendation program and dataset description

Our analysis focuses on the recommendation referral program run by a large retailer. The program rules were as follows. Each time a person purchases a book, music, or a movie he or she is given the option of sending emails recommending the item to friends. The first person to purchase the same item through a referral link in the email gets a 10% discount. When this happens the sender of the recommendation receives a 10% credit on their purchase.

The following information is recorded for each recommendation

1. Sender Customer ID (shadowed)
2. Receiver Customer ID (shadowed)
3. Date of Sending
4. Purchase flag (*buy-bit*)
5. Purchase Date (error-prone due to asynchrony in the servers)

SYMBOL	DESCRIPTION
n_p	Number of products
N_s	Number of senders of recommendations
N_r	Number of recommendation receivers
N	Number of nodes, $N = N_s \cup N_r$
r_r	Number of recommendations
E	Number of edges, <i>i.e.</i> , unique pairs of nodes that exchanged recommendations
$buy\text{-}bit$	Whether a recommendation results in a purchase that received discount
b_b	Number of purchases with buy-bit turned on
$buy\text{-}edge$	If a node got a recommendation and then sent another one then it must have bought
b_e	Number of purchases as determined via buy-edges
N_c	Number of nodes in the largest weakly connected component
r_c	Number of recommendation in the largest component
E_c	Number of edges in largest component
cc	Fraction of nodes in largest connected component, $cc = 100N_c/N$
γ	Power law degree exponent, $p(d) \propto d^{-\gamma}$
N_t	Size of the cascade at time t
p_t	Probability of a recommendation causing a purchase
r_{p1}	Average number of reviews per product in 2001–2003
v_{av}	Average star rating
c_{av}	Average number of people recommending a product
p_m	Median product price
b_r	Purchases per recommender, $b_r = (b_b + b_e)/r$

Table 6.1: Table of symbols.

6. Product identifier

7. Price

The recommendation dataset consists of 15,646,121 recommendations made among 3,943,084 distinct users. The data was collected from June 5 2001 to May 16 2003. In total, 548,523 products were recommended, 99% of them belonging to 4 main product groups: Books, DVDs, Music and Videos. In addition to recommendation data, we also crawled the retailer’s website to obtain product categories, reviews and ratings for all products. Of the products in our data set, 5813 (1%) were discontinued (the retailer no longer provided any information about them).

Although the data gives us a detailed and accurate view of recommendation dynamics, it does have its limitations. The only indication of the success of a recommendation is the observation of the recipient purchasing the product through the same vendor. We have no way of knowing if the person had decided instead to purchase elsewhere, borrow, or otherwise obtain the product. The delivery of the recommendation is also somewhat different from one person simply telling another about a product they enjoy, possibly in the context of a broader discussion of similar products. The recommendation is received as a form email including information about the discount program. Someone reading the email might consider it spam, or at least deem it less important than a recommendation given in the context of a conversation. The recipient may also doubt whether the friend is recommending the product because they think the recipient might enjoy it, or are simply trying to get a discount for themselves. Finally, because the recommendation takes place before the recommender receives the product, it might not be based on a direct observation of

the product. Nevertheless, we believe that these recommendation networks are reflective of the nature of word of mouth advertising, and give us key insights into the influence of social networks on purchasing decisions.

6.3.2 Identifying successful recommendations

For each recommendation, the dataset includes information about the recommended product, sender and received or the recommendation, and most importantly, the success of recommendation. See section 6.3.1 for more details.

We represent this data set as a directed multi graph. The nodes represent customers, and a directed edge contains all the information about the recommendation. The edge (i, j, p, t) indicates that i recommended product p to customer j at time t . Note that as there can be multiple recommendations between the persons (even on the same product) there can be multiple edges between two nodes.

The typical process generating edges in the recommendation network is as follows: a node i first buys a product p at time t and then it recommends it to nodes j_1, \dots, j_n . The j nodes can then buy the product and further recommend it. The only way for a node to recommend a product is to first buy it. Note that even if all nodes j buy a product, only the edge to the node j_k that first made the purchase (within a week after the recommendation) will be marked by a *buy-bit*. Because the buy-bit is set only for the first person who acts on a recommendation, we identify additional purchases by the presence of outgoing recommendations for a person, since all recommendations must be *preceded* by a purchase. We call this type of evidence of purchase a *buy-edge*. Note that buy-edges provide only a lower bound on the total number of purchases without discounts. It is possible for a customer to not be the first to act on a recommendation and also to not recommend the product to others. Unfortunately, this was not recorded in the data set. We consider, however, the buy-bits and buy-edges as proxies for the total number of purchases through recommendations.

As mentioned above the first buyer only gets a discount (the buy-bit is turned on) if the purchase is made within one week of the recommendation. In order to account for as many purchases as possible, we consider all purchases where the recommendation preceded the purchase (buy-edge) regardless of the time difference between the two events.

To avoid confusion we will refer to edges in a multi graph as recommendations (or multi-edges) — there can be more than one recommendation between a pair of nodes. We will use the term edge (or unique edge) to refer to edges in the usual sense, *i.e.*, there is only one edge between a pair of people. And, to get from recommendations to edges we create an edge between a pair of people if they exchanged at least one recommendation.

6.3.3 Properties of the recommendation network

For each product group we took recommendations on all products from the group and created a network. Table 6.2 shows the sizes of various product group recommendation networks with n_p being the total number of products in the product group, N the total number of nodes spanned by the group recommendation network, and r_r the number of recommendations (there can be multiple recommendations between two nodes). Column E shows the number of (unique) edges – disregarding multiple recommendations

Group	n_p	N	r_r	E	b_b	b_e
Book	103,161	2,863,977	5,741,611	2,097,809	65,344	17,769
DVD	19,829	805,285	8,180,393	962,341	17,232	58,189
Music	393,598	794,148	1,443,847	585,738	7,837	2,739
Video	26,131	239,583	280,270	160,683	909	467
Full network	542,719	3,943,084	15,646,121	3,153,676	91,322	79,164

Table 6.2: Product group recommendation statistics. n_p : number of products, N : number of nodes, r_r : number of recommendations, E : number of edges, b_b : number of buy bits, b_e : number of buy edges.

Group	N_c	r_c	E_c	b_{bc}	b_{ec}
Book	53,681	933,988	184,188	1,919	1,921
DVD	39,699	6,903,087	442,747	6,199	41,744
Music	22,044	295,543	82,844	348	456
Video	4,964	23,555	15,331	2	74
Full network	100,460	8,283,753	521,803	8,468	44,195

Table 6.3: Statistics for the largest connected component of each product group. N_c : number of nodes in largest connected component, r_c : number recommendations in the component, E_c : number of edges in the component, b_{bc} : number of buy bits, b_{ec} : number of buy edges in the largest connected component, and b_{bc} and b_{ec} are the number of purchase through a buy-bit and a buy-edge, respectively.

between the same source and recipient (*i.e.*, number of pairs of people that exchanged at least one recommendation).

In terms of the number of different items, there are by far the most music CDs, followed by books and videos. There is a surprisingly small number of DVD titles. On the other hand, DVDs account for more half of all recommendations in the dataset. The DVD network is also the most dense, having about 10 recommendations per node, while books and music have about 2 recommendations per node and videos have only a bit more than 1 recommendation per node.

Music recommendations reached about the same number of people as DVDs but used more than 5 times fewer recommendations to achieve the same coverage of the nodes. Book recommendations reached by far the most people – 2.8 million. Notice that all networks have a very small number of unique edges. For books, videos and music the number of unique edges is smaller than the number of nodes – this suggests that the networks are highly disconnected [Erdős and Rényi, 1960].

Back to table 6.2: given the total number of recommendations r_r and purchases ($b_b + b_e$) influenced by recommendations we can estimate how many recommendations need to be independently sent over the network to induce a new purchase. Using this metric books have the most influential recommendations followed by DVDs and music. For books one out of 69 recommendations resulted in a purchase. For DVDs it increases to 108 recommendations per purchase and further increases to 136 for music and 203 for video.

Table 6.3 gives more insight into the structure of the largest connected component of each product group’s recommendation network. We performed the same measurements as in table 6.2 with the difference being that we did not use the whole network but only its largest weakly connected component. The table shows

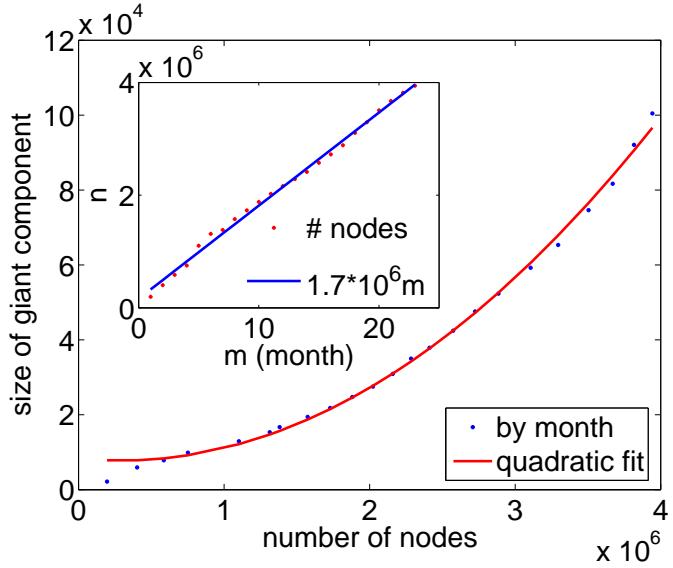


Figure 6.1: (a) The size of the largest connected component of customers over time. The inset shows the linear growth in the number of customers N over time.

the number of nodes N , the number of recommendations r_c , and the number of (unique) edges E_c in the largest component. The last two columns (b_{bc} and b_{ec}) show the number of purchases resulting in a discount (buy-bit, b_{bc}) and the number of purchases through buy-edges (b_{ec}) in the largest connected component.

First, notice that the largest connected components are very small. DVDs have the largest - containing 4.9% of the nodes, books have the smallest at 1.78%. One would also expect that the fraction of the recommendations in the largest component would be proportional to its size. We notice that this is not the case. For example, the largest component in the full recommendation network contains 2.54% of the nodes and 52.9% of all recommendations, which is the result of heavy bias in DVD recommendations. Breaking this down by product categories we see that for DVDs 84.3% of the recommendations are in the largest component (which contains 4.9% of all DVD nodes), vs. 16.3% for book recommendations (component size 1.79%), 20.5% for music recommendations (component size 2.77%), and 8.4% for video recommendations (component size 2.1%). This shows that the dynamic in the largest component is very much different from the rest of the network. Especially for DVDs we can see that a very small fraction of users generated most of the recommendations.

6.3.4 Recommendation network over time

The recommendations that occurred were exchanged over an existing underlying social network. In the real world, it is estimated that any two people on the globe are connected via a short chain of acquaintances - popularly known as the small world phenomenon [Travers and Milgram, 1969]. We examined whether the edges formed by aggregating recommendations over all products would similarly yield a small world network, even though they represent only a small fraction of a person's complete social network. We measured the growth of the largest weakly connected component over time, shown in Figure 6.1. Within the weakly connected component, any node can be reached from any other node by traversing (undirected)

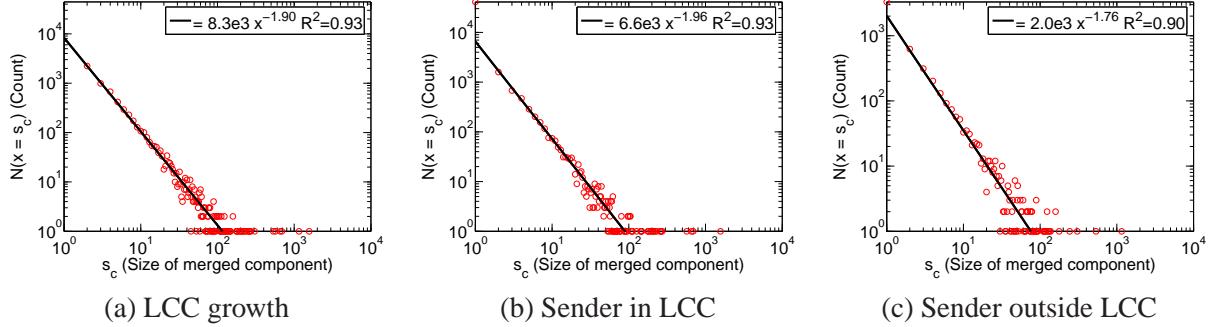


Figure 6.2: Growth of the largest connected component (LCC). (a) the distribution of sizes of components when they are merged into the largest connected component. (b) same as (a), but restricted to cases when a member of the LCC sends a recommendation to someone outside the largest component. (c) a sender outside the largest component sends a recommendation to a member of the component.

edges. For example, if u recommended product x to v , and w recommended product y to v , then u and w are linked through one intermediary and thus belong to the same weakly connected component. Note that connected components do not necessarily correspond to communities (clusters) which we often think of as densely linked parts of the networks. Nodes belong to same component if they can reach each other via an undirected path regardless of how densely they are linked.

Figure 6.1 shows the size of the largest connected component, as a fraction of the total network. The largest component is very small over all time. Even though we compose the network using all the recommendations in the dataset, the largest connected component contains less than 2.5% (100,420) of the nodes, and the second largest component has only 600 nodes. Still, some smaller communities, numbering in the tens of thousands of purchasers of DVDs in categories such as westerns, classics and Japanese animated films (anime), had connected components spanning about 20% of their members.

The insert in figure 6.1 shows the growth of the customer base over time. Surprisingly it was linear, adding on average 165,000 new users each month, which is an indication that the service itself was not spreading epidemically. Further evidence of non-viral spread is provided by the relatively high percentage (94%) of users who made their first recommendation without having previously received one.

Growth of the largest connected component

Next, we examine the growth of the largest connected component (LCC). In figure 6.1 we saw that the largest component seems to grow quadratically over time, but at the end of the data collection period is still very small, *i.e.*, only 2.5% of the nodes belong to largest weakly connected component. Here we are not interested in how fast the largest component grows over time but rather how big other components are when they get merged into the largest component. Also, since our graph is directed we are interested in determining whether smaller components become attached to the largest component by a recommendation sent from inside of the largest component. One can think of these recommendations as being tentacles reaching out of largest component to attach smaller components. The other possibility is that the recommendation comes from a node outside the component to a member of the largest component and thus the initiative to attach comes from outside the largest component.

We look at whether the largest component grows gradually, adding nodes one by one as the members send out more recommendations, or whether a new recommendation might act as a bridge to a component consisting of several nodes who are already linked by their previous recommendations. To this end we measure the distribution of a component's size when it gets merged to the largest weakly connected component.

We operate under the following setting. Recommendations are arriving over time one by one creating edges between the nodes of the network. As more edges are being added the size of largest connected component grows. We keep track of the currently largest component, and measure how big the separate components are when they get attached to the largest component.

Figure 6.2(a) shows the distribution of merged connected component (CC) sizes. On the x-axis we plot the component size (number of nodes N) and on the y-axis the number of components of size N that were merged over time with the largest component. We see that a majority of the time a single node (component of size 1) merged with the currently largest component. On the other extreme is the case when a component of 1,568 nodes merged with the largest component.

Interestingly, out of all merged components, in 77% of the cases the source of the recommendation comes from inside the largest component, while in the remaining 23% of the cases it is the smaller component that attaches itself to the largest one. Figure 6.2(b) shows the distribution of component sizes only for the case when the sender of the recommendation was a member of the largest component, *i.e.*, the small component was attached from the largest component. Lastly, Figure 6.2(c) shows the distribution for the opposite case when the sender of the recommendation was not a member of the largest component, *i.e.*, the small component attached itself to the largest.

Also notice that in all cases the distribution of merged component sizes follows a heavy-tailed distribution. We fit a power law distribution and note the power law exponent of 1.90 (fig. 6.2(a)) when considering all merged components. Limiting the analysis to the cases where the source of the edge that attached a small component to the largest is in the largest component we obtain power law exponent of 1.96 (fig. 6.2(b)), and when the edge originated from the small component to attached it to the largest, the power law exponent is 1.76. This shows that even though in most cases the LCC absorbs the small component, we see that components that attach themselves to the LCC tend to be larger (smaller power law exponent) than those attracted by the LCC. This means that the component sometimes grows a bit before it attaches itself to the largest component. Intuitively, an individual node can get attached to the largest component simply by passively receiving a recommendation. But if it is the outside node that sends a recommendation to someone in the giant component, it is already an active recommender and could therefore have recommended to several others previously, thus forming a slightly bigger component that is then merged.

From these experiments we see that the largest component is very active, adding smaller components by generating new recommendations. Most of the time these newly merged components are quite small, but occasionally sizable components are attached.

6.3.5 Preliminary observations and discussion

Even with these simple counts and experiments we can already make a few observations. It seems that some people got quite heavily involved in the recommendation program, and that they tended to recommend a large number of products to the same set of friends (since the number of unique edges is so small as shown on table 6.2). This means that people tend to buy more DVDs and also like to recommend

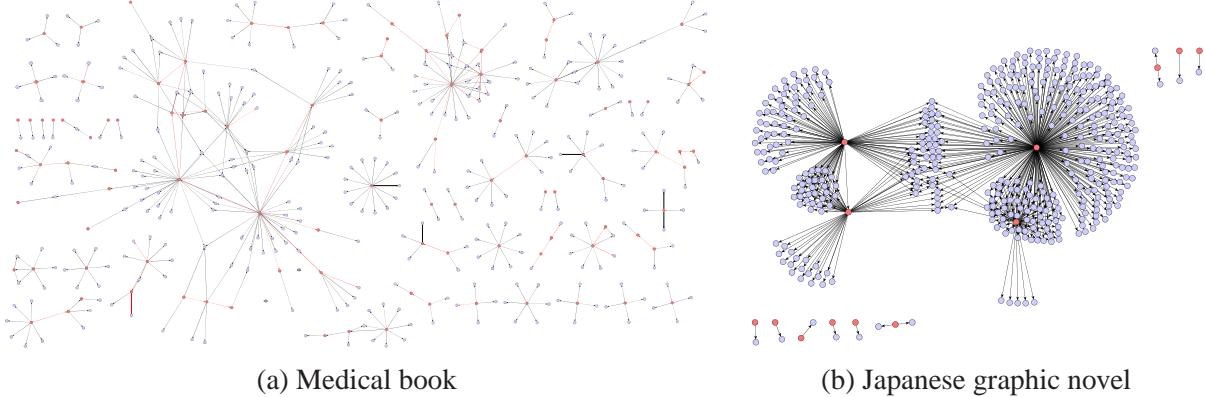


Figure 6.3: Examples of two product recommendation networks: (a) First aid study guide *First Aid for the USMLE Step*, (b) Japanese graphic novel (manga) *Oh My Goddess!: Mara Strikes Back*.

them to their friends, while they seem to be more conservative with books. One possible reason is that a book is a bigger time investment than a DVD: one usually needs several days to read a book, while a DVD can be viewed in a single evening. Another factor may be how informed the customer is about the product. DVDs, while fewer in number, are more heavily advertised on TV, billboards, and movie theater previews. Furthermore, it is possible that a customer has already watched a movie and is adding the DVD to their collection. This could make them more confident in sending recommendations before viewing the purchased DVD.

One external factor which may be affecting the recommendation patterns for DVDs is the existence of referral websites (www.dvdtalk.com). On these websites people, who want to buy a DVD and get a discount, would ask for recommendations. This way there would be recommendations made between people who don't really know each other but rather have an economic incentive to cooperate.

In effect, the viral marketing program is altering, albeit briefly and most likely unintentionally, the structure of the social network it is spreading on. We were not able to find similar referral sharing sites for books or CDs.

6.4 Propagation of recommendations

6.4.1 Forward recommendations

Not all people who accept a recommendation by making a purchase also decide to give recommendations. In estimating what fraction of people that purchase also decide to recommend forward, we can only use the nodes with purchases that resulted in a discount. Table 6.4 shows that only about a third of the people that purchase also recommend the product forward. The ratio of forward recommendations is much higher for DVDs than for other kinds of products. Videos also have a higher ratio of forward recommendations, while books have the lowest. This shows that people are most keen on recommending movies, possibly for the above mentioned reasons, while more conservative when recommending books and music.

Figure 6.4 shows the cumulative out-degree distribution, that is the number of people who sent out at least k_p recommendations, for a product. We fit a power law to all but the tail of the distribution. Also,

Group	Number of nodes		
	Purchases	Forward	Percent
Book	65,391	15,769	24.2
DVD	16,459	7,336	44.6
Music	7,843	1,824	23.3
Video	909	250	27.6
Total	90,602	25,179	27.8

Table 6.4: Fraction of people that purchase and also recommend forward. *Purchases*: number of nodes that purchased as a result of receiving a recommendation. *Forward*: nodes that purchased and then also recommended the product to others.

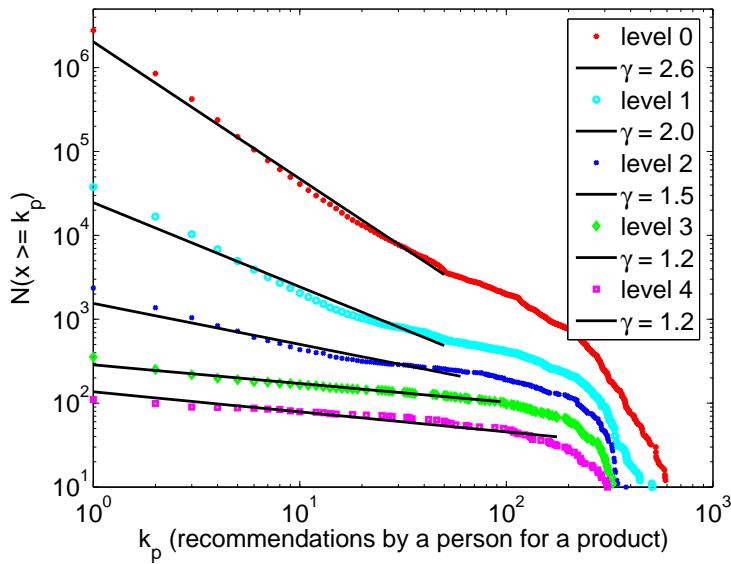


Figure 6.4: The number of recommendations sent by a user with each curve representing a different depth of the user in the recommendation chain. A power law exponent γ is fitted to all but the tail, which shows an exponential drop-off at around 100 recommendations sent. This drop-off is consistent across all depth levels, and may reflect either a natural disinclination to send recommendation to over a hundred people, or a technical issue that might have made it more inconvenient to do so. The fitted lines follow the order of the level number (*i.e.*, top line corresponds to level 0 and bottom to level 4).

notice the exponential decay in the tail of the distribution which could be, among other reasons, attributed to the finite time horizon of our dataset. (Note that the reasons for exponential decay here are different than in Chapter 4 where we investigated microscopic network evolution. There the power law exponents remained constant and the exponential decay factor got stronger as node degree increased.)

The figure 6.4 shows that the deeper an individual is in the cascade, if they choose to make recommendations, they tend to recommend to a greater number of people on average (the fitted line has smaller slope γ , *i.e.*, the distribution has higher variance). This effect is probably due to only very heavily recommended products producing large enough cascades to reach a certain depth. We also observe, as is shown in Table 6.5, that the probability of an individual making a recommendation at all (which can only occur if they make a purchase), declines after an initial increase as one gets deeper into the cascade.

level	prob. buy & forward	average out-degree
0	N/A	1.99
1	0.0069	5.34
2	0.0149	24.43
3	0.0115	72.79
4	0.0082	111.75

Table 6.5: Statistics about individuals at different levels of the cascade.

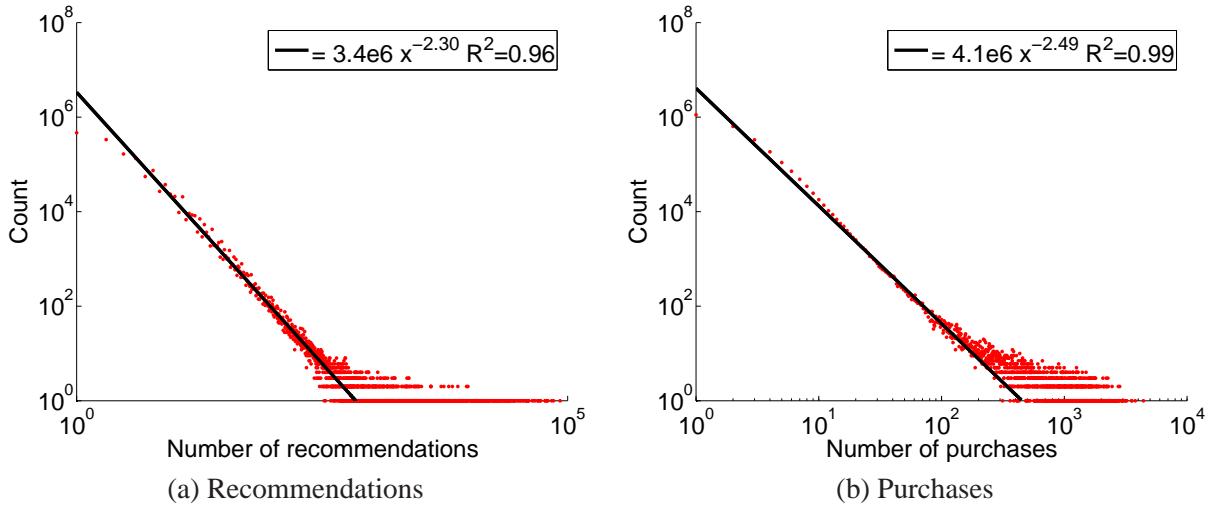


Figure 6.5: Distribution of the number of recommendations and number of purchases made by a customer.

6.4.2 Identifying cascades

As customers continue forwarding recommendations, they contribute to the formation of cascades. In order to identify cascades, *i.e.*, the “causal” propagation of recommendations, we track *successful recommendations* as they influence purchases and further recommendations. We define a recommendation to be successful if it reached a node before its *first* purchase. We consider only the first purchase of an item, because there are many cases when a person made multiple purchases of the same product, and in between those purchases she may have received new recommendations. In this case one cannot conclude that recommendations following the first purchase influenced the later purchases.

Each cascade is a network consisting of customers (nodes) who purchased the same product as a result of each other’s recommendations (edges). We delete *late recommendations* — all incoming recommendations that happened after the first purchase of the product. This way we make the network *time increasing* or *causal* — for each node all incoming edges (recommendations) occurred before all outgoing edges. Now each connected component represents a time obeying propagation of recommendations.

Figure 6.3 shows two typical product recommendation networks: (a) a medical study guide and (b) a Japanese graphic novel. Throughout the dataset we observe very similar patterns. Most product recommendation networks consist of a large number of small disconnected components where we do not observe cascades. Then there is usually a small number of relatively small components with recommendations

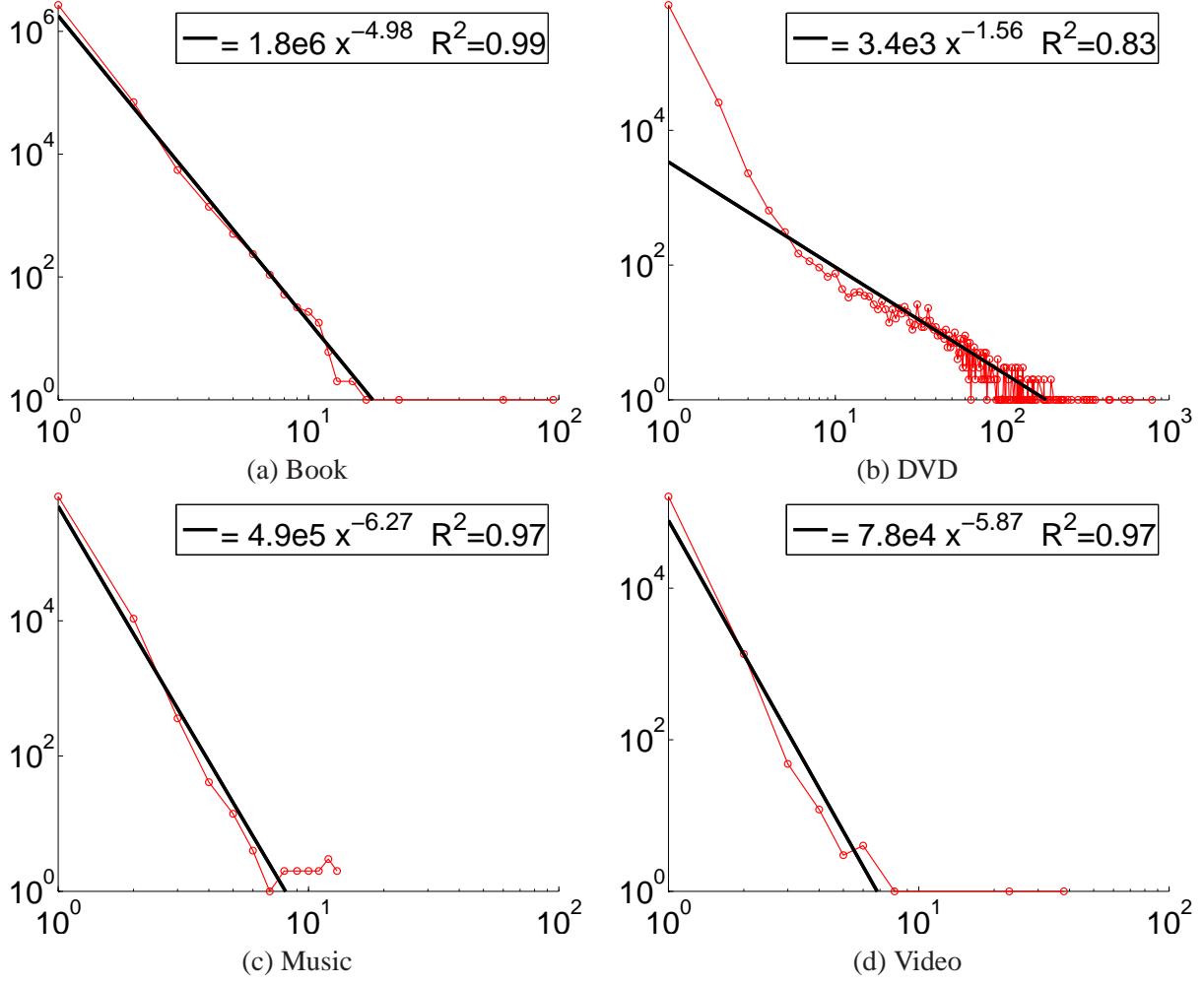


Figure 6.6: Size distribution of cascades (size of cascade vs. count). Bold line presents a power-fit.

successfully propagating. This observation is reflected in the heavy tailed distribution of cascade sizes (see figure 6.6), having a power law exponent close to 1 for DVDs in particular. We determined the power law exponent by fitting a line on log-log scales using the least squares method.

We also notice bursts of recommendations (figure 6.3(b)). Some nodes recommend to many friends, forming a star like pattern. Figure 6.5 shows the distribution of the recommendations and purchases made by a single node in the recommendation network. Notice the power law distributions and long flat tails. The most active customer made 83,729 recommendations and purchased 4,416 different items. Finally, we also sometimes observe “collisions”, where nodes receive recommendations from two or more sources. A detailed enumeration and analysis of observed topological cascade patterns for this dataset is made in section 6.9.

Last, we examine the number of exchanged recommendations between a pair of people in figure 6.7. Overall, 39% of pairs of people exchanged just a single recommendation. This number decreases for DVDs to 37%, and increases for books to 45%. The distribution of the number of exchanged recommendations follows a heavy tailed distribution. To get a better understanding of the distributions we show the power

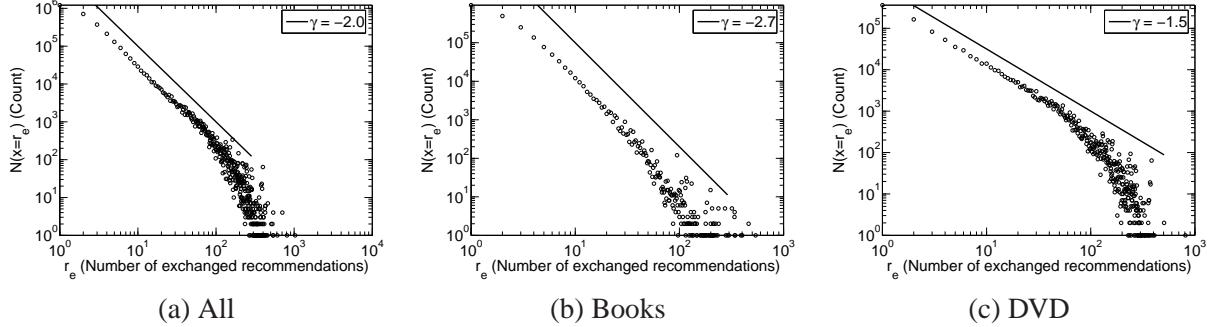


Figure 6.7: Distribution of the number of exchanged recommendations between pairs of people.

law decay lines. Notice that one gets much stronger decay exponent (distribution has weaker tail) of -2.7 for books and a very shallow power law exponent of -1.5 for DVDs. This means that even a pair of people exchanges more DVD than book recommendations.

6.4.3 The recommendation propagation model

A simple model can help explain how the wide variance we observe in the number of recommendations made by individuals can lead to power laws in cascade sizes (figure 6.6). The model assumes that each recipient of a recommendation will forward it to others if its value exceeds an arbitrary threshold that the individual sets for herself. Since exceeding this value is a probabilistic event, let's call p_t the probability that at time step t the recommendation exceeds the threshold. In that case the number of recommendations N_{t+1} at time $(t + 1)$ is given in terms of the number of recommendations at an earlier time by

$$N_{t+1} = (1 + p_t)N_t \quad (6.1)$$

where the probability p_t is defined over the unit interval.

Notice that, because of the probabilistic nature of the threshold being exceeded, one can only compute the final distribution of recommendation chain lengths, which we now proceed to do.

Subtracting from both sides of this equation the term N_t and diving by it we obtain

$$\frac{N_{(t+1)} - N_t}{N_t} = p_t \quad (6.2)$$

Summing both sides from the initial time to some very large time T and assuming that for long times the numerator is smaller than the denominator (a reasonable assumption) we get, up to a unit constant

$$\int \frac{dN}{N} \approx \sum \frac{N_{(t+1)} - N_t}{N_t} = \sum p_t \quad (6.3)$$

The left hand integral is just $\ln(N)$, and the right hand side is a sum of random variables, which in the limit of a very large uncorrelated number of recommendations is normally distributed (central limit theorem).

This observation was first made by Gibrat [Gibrat, 1931] to model the growth rates of firms and is known as the Law of Proportional Effect or simply Gibrat's Law.

So, this means that the logarithm of the number of messages is normally distributed. Or equivalently, the number of messages passed is log-normally distributed. So, the probability density for N is given by

$$P(N) = \frac{1}{N\sqrt{2\pi\sigma^2}} \exp \frac{-(\ln(N) - \mu)^2}{2\sigma^2} \quad (6.4)$$

which, for large variances describes a behavior whereby the typical number of recommendations is small (the mode of the distribution) but there are unlikely events of large chains of recommendations which are also observable.

Furthermore, for large variances, the lognormal distribution can behave like a power law for a range of values. In order to see this, take the logarithms on both sides of the equation (equivalent to a log-log plot) and one obtains

$$\ln(P(N)) = -\ln(N) - \ln(\sqrt{2\pi\sigma^2}) - \frac{(\ln(N) - \mu)^2}{2\sigma^2} \quad (6.5)$$

So, for large σ , the last term of the right hand side goes to zero, and since the second term is a constant one obtains a power law behavior with exponent value of minus one [Bi et al., 2001]. There are other models which produce power law distributions of cascade sizes, but we present ours for its simplicity, since it does not depend on network topology [Gruhl et al., 2004] or critical thresholds in the probability of a recommendation being accepted [Watts, 2002]. Also, similar derivation of lognormal distribution can be found in [Johnson et al., 1994] and is also known as the “law of proportional effect”.

6.5 Success of Recommendations

So far we only looked into the aggregate statistics of the recommendation network. Next, we ask questions about the effectiveness of recommendations in the recommendation network itself. First, we analyze the probability of purchasing as one gets more and more recommendations. Next, we measure recommendation effectiveness as two people exchange more and more recommendations. Lastly, we observe the recommendation network from the perspective of the sender of the recommendation. Does a node that makes more recommendations also influence more purchases?

6.5.1 Human adoption curve: the probability of buying versus number of incoming recommendations

First, we examine how the probability of purchasing changes as one gets more and more recommendations. One would expect that a person is more likely to buy a product if she gets more recommendations. On the other hand one would also think that there is a saturation point – if a person hasn't bought a product after a number of recommendations, they are not likely to change their minds after receiving even more of them. So, how many recommendations are too many?

Figure 6.8 shows the probability of purchasing a product as a function of the number of incoming recommendations on the product. Because we exclude late recommendations, those that were received after the

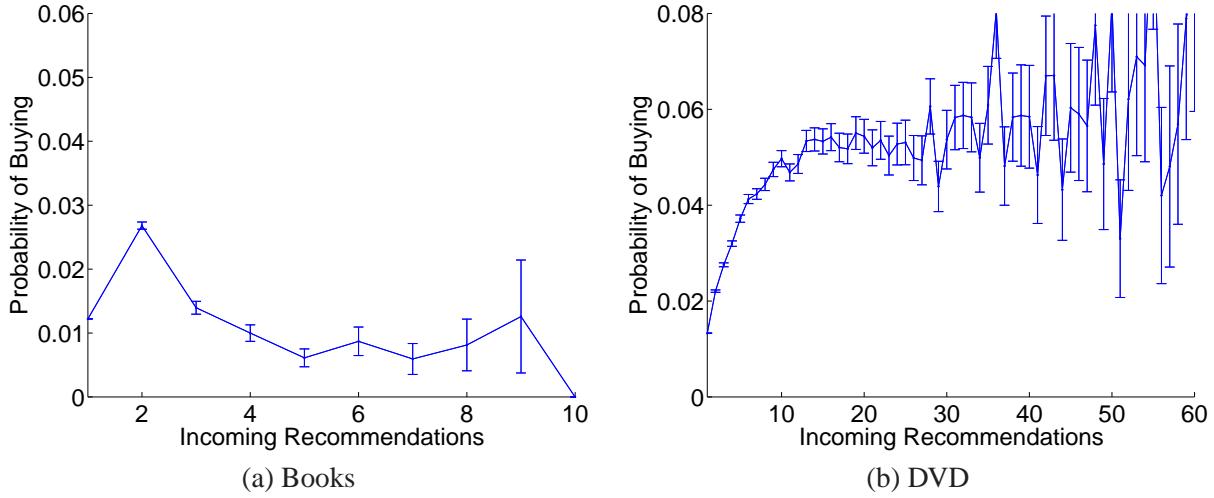


Figure 6.8: Probability of buying a book (DVD) given a number of incoming recommendations. This shows the human adoption curve has the diminishing returns property.

purchase, an individual counts as having received three recommendations only if they did not make a purchase after the first two, and either purchased or did not receive further recommendations after receiving the third one. As we move to higher numbers of incoming recommendations, the number of observations drops rapidly. For example, there were 5 million cases with 1 incoming recommendation on a book, and only 58 cases where a person got 20 incoming recommendations on a particular book. The maximum was 30 incoming recommendations. For these reasons we cut-off the plot when the number of observations becomes too small and the error bars too large.

We calculate the purchase probabilities and the standard errors of the estimates which we use to plot the error bars in the following way. We regard each point as a binomial random variable. Given the number of observations n , let m be the number of successes, and k ($k = n - m$) the number of failures. In our case, m is the number of people that first purchased a product after receiving r recommendations on it, and k is the number of people that received the total of r recommendations on a product (till the end of the dataset) but did purchase it, then the estimated probability of purchasing is $\hat{p} = m/n$ and the standard error $s_{\hat{p}}$ of estimate \hat{p} is $s_{\hat{p}} = \sqrt{p(1-p)/n}$.

Figure 6.8(a) shows that, overall, book recommendations are rarely followed. Even more surprisingly, as more and more recommendations are received, their success decreases. We observe a peak in probability of buying at 2 incoming recommendations and then a slow drop. This implies that if a person doesn't buy a book after the first recommendation, but receives another, they are more likely to be persuaded by the second recommendation. But thereafter, they are less likely to respond to additional recommendations, possibly because they perceive them as spam, are less susceptible to others' opinions, have a strong opinion on the particular product, or have a different means of accessing it.

For DVDs (figure 6.8(b)) we observe a saturation around 10 incoming recommendations. This means that with each additional recommendation, a person is more and more likely to be persuaded - up to a point. After a person gets 10 recommendations on a particular DVD, their probability of buying does not increase anymore. The number of observations is 2.5 million at 1 incoming recommendation and 100 at 60 incoming recommendations. The maximal number of received recommendations is 172 (and that person

did not buy), but someone purchased a DVD after 169 receiving recommendations. The different patterns between book and DVD recommendations may be a result of the recommendation exchange websites for DVDs. Someone receiving many DVD recommendations may have signed up to receive them for a product they intended to purchase, and hence a greater number of received recommendations corresponds to a higher likelihood of purchase (up to a point).

6.5.2 Success of subsequent recommendations

Next, we analyze how the effectiveness of recommendations changes as one received more and more recommendations from the same person. A large number of exchanged recommendations can be a sign of trust and influence, but a sender of too many recommendations can be perceived as a spammer. A person who recommends only a few products will have her friends' attention, but one who floods her friends with all sorts of recommendations will start to lose her influence.

We measure the effectiveness of recommendations as a function of the total number of previously received recommendations from a particular node. We thus measure how spending changes over time, where time is measured in the number of received recommendations.

We construct the experiment in the following way. For every recommendation r on some product p between nodes u and v , we first determine how many recommendations node u received from v before getting r . Then we check whether v , the recipient of recommendation, purchased p after the recommendation r arrived. If so, we count the recommendation as successful since it influenced the purchase. This way we can calculate the recommendation success rate as more recommendations were exchanged. For the experiment we consider only node pairs (u, v) , where there were at least a total of 10 recommendations sent from u to v . We perform the experiment using only recommendations from the same product group.

We decided to set a lower limit on the number of exchanged recommendations so that we can measure how the effectiveness of recommendations changes as the *same* two people exchange more and more recommendations. Considering all pairs of people would heavily bias our findings since most pairs exchange just a few or even just a single recommendation. Using the data from figure 6.7 we see that 91% of pairs of people that exchange at least 1 recommendation exchange less than 10. For books this number increases to 96%, and for DVDs it is even smaller (81%). In the DVD network there are 182 thousand pairs that exchanged more than 10 recommendations, and 70 thousand for the book network.

Figure 6.9 shows the probability of buying as a function of the total number of received recommendations from a particular person up to that point. One can think of x-axis as measuring time where the unit is the number of received recommendations from a particular person.

For books we observe that the effectiveness of recommendation remains about constant up to 3 exchanged recommendations. As the number of exchanged recommendations increases, the probability of buying starts to decrease to about half of the original value and then levels off. For DVDs we observe an immediate and consistent drop. We performed the experiment also for video and music, but the number of observations was too low and the measurements were noisy. This experiment shows that recommendations start to lose effect after more than two or three are passed between two people. Also, notice that the effectiveness of book recommendations show in Figure 6.9(a) decays much more slowly than that of DVD recommendations (Figure 6.9(b)), flattening out at around 20 recommendations, compared to around 10 DVD exchanged recommendations.

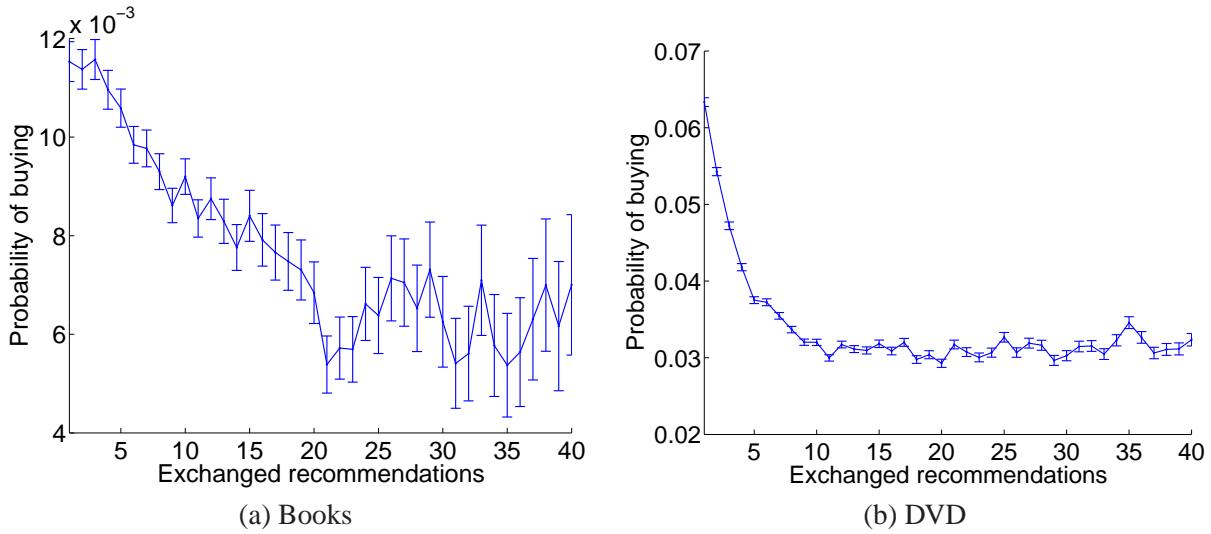


Figure 6.9: The effectiveness of recommendations with the number of received recommendations.

This result has important implications for viral marketing practitioners as it shows that by providing too much incentive for people to recommend to one another can weaken the very social network links that the marketer is intending to exploit.

6.5.3 Success of outgoing recommendations

In previous sections we examined the data from the viewpoint of the receiver of the recommendation. Now we look from the viewpoint of the sender. The two interesting questions are: how does the probability of getting a 10% credit change with the number of outgoing recommendations; and given a number of outgoing recommendations, how many purchases will they influence?

One would expect that recommendations would be the most effective when recommended to the right subset of friends. If one is very selective and recommends to too few friends, then the chances of success are slim. On the other hand, recommending to everyone and spamming them with recommendations may have limited returns as well.

The top row of figure 6.10 shows how the average number of purchases changes with the number of outgoing recommendations. For books, music, and VHS videos the number of purchases soon saturates: purchases grow fast up to around 10 outgoing recommendations and then the trend either slows down or starts to drop. DVDs exhibit different behavior, with the expected number of purchases increasing throughout.

These results are even more interesting since the receiver of the recommendation does not know how many other people also received the recommendation. Thus the plots of figure 6.10 show that there are interesting dependencies between the product characteristics and the recommender that manifest through the number of recommendations sent. It could be the case that widely recommended products are not suitable for viral marketing (we find something similar in section 6.8.2), or that the recommender did not put too much thought into who to send the recommendation to, or simply that people soon start to ignore mass recommenders.

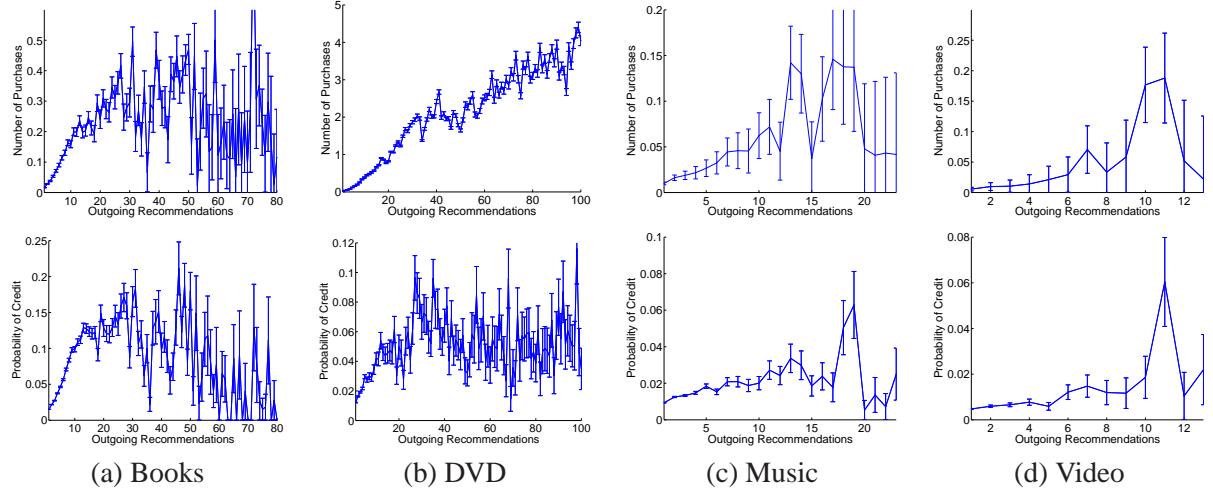


Figure 6.10: Top row: Number of resulting purchases given a number of outgoing recommendations.
Bottom row: Probability of getting a credit given a number of outgoing recommendations.

Plotting the probability of getting a 10% credit as a function of the number of outgoing recommendations, as in the bottom row of figure 6.10, we see that the success of DVD recommendations saturates as well, while books, videos and music have qualitatively similar trends. The difference in the curves for DVD recommendations points to the presence of collisions in the dense DVD network, which has 10 recommendations per node and around 400 per product — an order of magnitude more than other product groups. This means that many different individuals are recommending to the same person, and after that person makes a purchase, even though all of them made a ‘successful recommendation’ by our definition, only one of them receives a credit.

6.5.4 Success of incoming recommendations

The collisions of recommendations are a dominant feature of the DVD recommendation network. Book recommendations have the highest chance of getting a credit, but DVD recommendations cause the most purchases. So far it seems people are very keen on recommending various DVDs, while very conservative on recommending books. But how does the behavior of customers change as they get more involved into the recommendation network? We would expect that most of the people are not heavily involved, so their probability of buying is not high. In the extreme case we expect to find people who buy almost everything they get recommendations on.

There are two ways to measure the involvedness of a person in the network: by the total number of incoming recommendations (on all products) or the total number of different products they were recommended. For every purchase of a book at time t , we count the number of different books (DVDs, ...) the person received recommendations for before time t . As in all previous experiments we delete late recommendations, *i.e.*, recommendations that arrived after the first purchase of a product.

We show the probability of buying as a function of the number of different products recommended in Figure 6.11. Figure 6.12 plots the same data but with the total number of incoming recommendations on the x-axis. We calculate the error bars as described in section 6.5.1. The number of observations is large

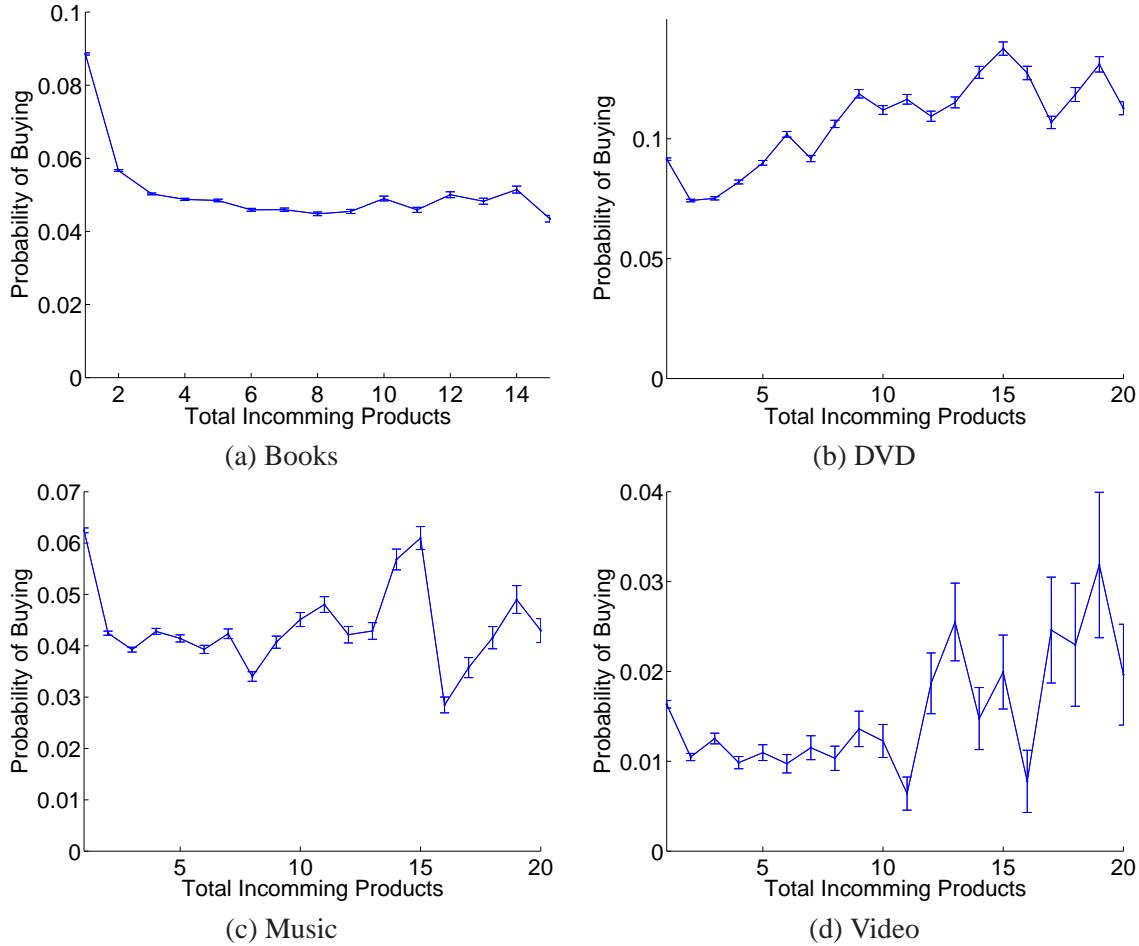


Figure 6.11: The probability of buying a product given a number of different products a node got recommendations on.

enough (error bars are sufficiently small) to draw conclusions about the trends observed in the figures. For example, there are more than 15,000 users that had 15 incoming DVD recommendations.

Notice that trends are quite similar regardless of whether we measure how involved is the user in the network by counting the number of products recommended (figure 6.11) or the number of incoming recommendations (fig. 6.12).

We observe two distinct trends. For books and music (figures 6.11 and 6.12, (a) and (c)) the probability of buying is the highest when a person got recommendations on just 1 item, as the number of recommended products increases to 2 or more the probability of buying quickly decreases and then flattens.

Movies (DVDs and videos) exhibit different behavior (figure 6.11 and 6.12, (b) and (d)). A person is more likely to buy the more recommendations she gets. For DVDs the peak is at around 15 incoming products, while for videos there is no such peak – the probability remains fairly level. Interestingly for DVDs the distribution reaches its low at 2 and 3 items, while for videos it lies somewhere between 3 and 8 items. The results suggest that books and music buyers tend to be conservative and focused. On the other hand there are people who like to buy movies in general. One could hypothesize that buying a book is a larger

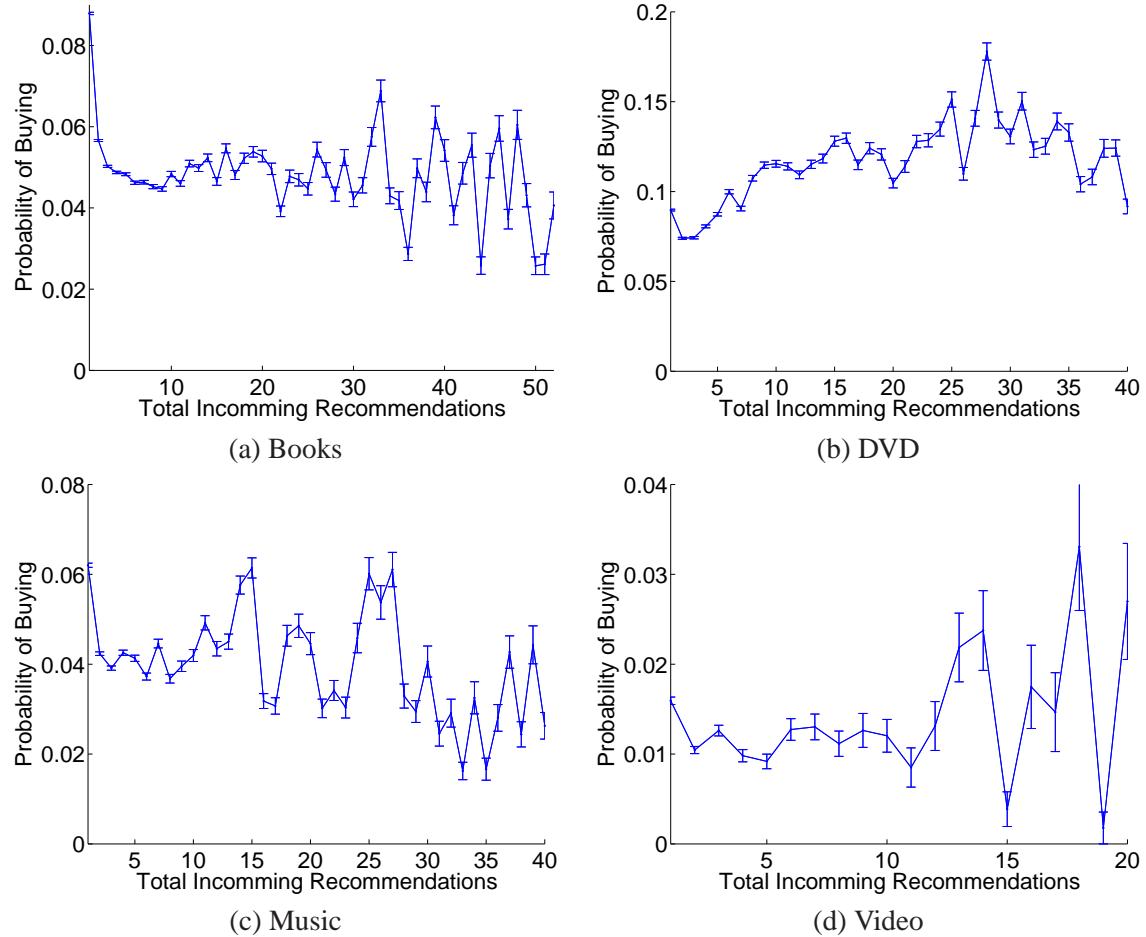


Figure 6.12: Probability of buying a product given a total number of incoming recommendations on all products.

investment of time and effort than buying a movie. One can finish a movie in an evening, while reading a book requires more effort. There are also many more book and music titles than movie titles.

The other difference between the book and music recommendations in comparison to movies are the recommendation referral websites where people could go to get recommendations. One could see these websites as recommendation subscription services – posting one’s email on a list results in a higher number of incoming recommendations. For movies, people with a high number of incoming recommendations “subscribed” to them and thus expected/wanted the recommendations. On the other hand people with high numbers of incoming book or music recommendations did not “sign up” for them, so they may perceive recommendations as spam and thus the influence of recommendations drops.

Another evidence of the existence of recommendations referral websites includes the DVD recommendation network degree distribution. The DVDs follow a power law degree distribution with an exception of a peak at out-degree 50. Other plots of DVD recommendation behavior also exhibited abnormalities at around 50 recommendations. These can be attributed to the recommendation referral websites.

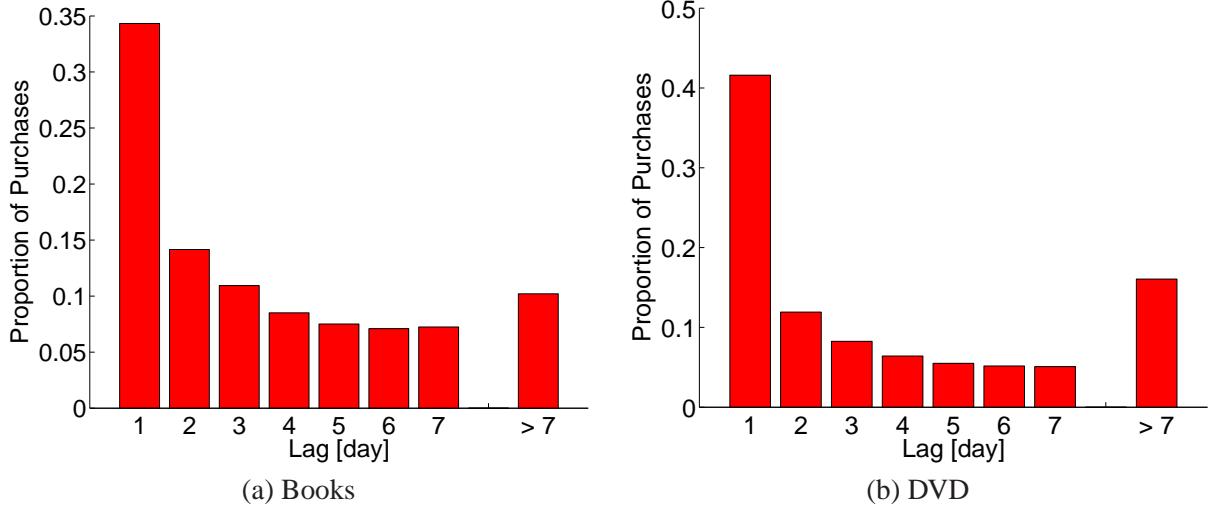


Figure 6.13: The time between the recommendation and the actual purchase. We use all purchases.

6.6 Timing of recommendations and purchases

The recommendation referral program encourages people to purchase as soon as possible after they get a recommendation, since this maximizes the probability of getting a discount. We study the time lag between the recommendation and the purchase of different product groups, effectively how long it takes a person to receive a recommendation, consider it, and act on it.

We present the histograms of the “thinking time”, *i.e.*, the difference between the time of purchase and the time the last recommendation was received for the product prior to the purchase (figure 6.13). We use a bin size of 1 day. Around 35%-40% of book and DVD purchases occurred within a day after the last recommendation was received. For DVDs 16% purchases occur more than a week after the last recommendation, while this drops to 10% for books. In contrast, if we consider the lag between the purchase and the *first* recommendation, only 23% of DVD purchases are made within a day, while the proportion stays the same for books. This reflects a greater likelihood for a person to receive multiple recommendations for a DVD than for a book. At the same time, DVD recommenders tend to send out many more recommendations, only one of which can result in a discount. Individuals then often miss their chance of a discount, which is reflected in the high ratio (78%) of recommended DVD purchases that did not get a discount (see table 6.2, columns b_b and b_e). In contrast, for books, only 21% of purchases through recommendations did not receive a discount.

We also measure the variation in intensity by time of day for three different activities in the recommendation system: recommendations (figure 6.14(a)), all purchases (figure 6.14(b)), and finally just the purchases which resulted in a discount (figure 6.14(c)). Each is given as a total count by hour of day.

The recommendations and purchases follow the same pattern. The only small difference is that purchases reach a sharper peak in the afternoon (after 3pm Pacific Time, 6pm Eastern time). This means that the willingness to recommend does not change with time, since about a constant fraction of purchases also result in recommendations sent (plots 6.14(a) and (b) follow the same shape).

The purchases that resulted in a discount (fig. 6.14(c)) look like a negative image of the first two figures. If recommendations would have no effect then plot (c) should follow the same shape as (a) and (b), since

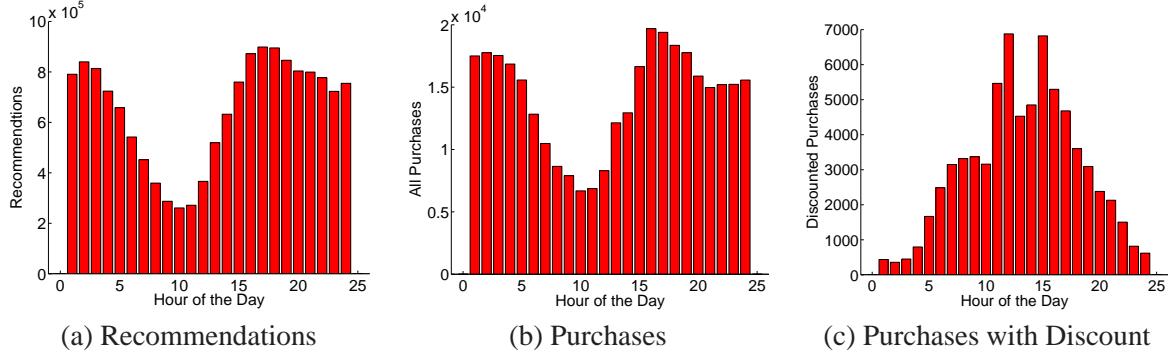


Figure 6.14: Time of day for purchases and recommendations. (a) shows the distribution of recommendations over the day, (b) shows all purchases and (c) shows only purchases that resulted in a discount.

a fraction of people that buy would become first buyers, *i.e.*, the more recommendations sent, the more first buyers and thus discounts. However, this does not seem to be the case. The number of purchases with discount is high when the number of purchases is small. This means that most of discounted purchases happened in the morning when the traffic (number of purchases/recommendations) on the retailer's website was low. This makes sense since most of the recommendations happened during the day, and if the person wanted to get the discount by being the first one to purchase, she had the highest chances when the traffic on the website was the lowest.

There are also other factors that come into play here. Assuming that recommendations are sent to people's personal (non-work) email addresses, then people probably check these email accounts for new email less regularly while at work. So checking personal email while at work and reacting to a recommendation would mean higher chances of getting a discount. Second, there are also network effects, *i.e.*, the more recommendations sent, the higher chance of recommendation collision, the lower chance of getting discount, since one competes with the larger set of people.

6.7 Recommendations and communities of interest

Social networks are a product of the contexts that bring people together. The context can be a shared interest in a particular topic or kind of a book. Sometimes there are circumstances, such as a specific job or religious affiliation, that would make people more likely to be interested in the same type of book or DVD. We first apply a community discovery algorithm to automatically detect communities of individuals who exchange recommendations with one another and to identify the kinds of products each community prefers. We then compare the effectiveness of recommendations across book categories, showing that books on different subjects have varying success rates.

6.7.1 Communities and purchases

In aggregating all recommendations between any two individuals in Section 6.3.4 we showed that the network consists of one large component, containing a little over 100,000 customers, and many smaller

components, the largest of which has 634 customers. However, knowing that a hundred thousand customers are linked together in a large network does not reveal whether a product in a particular category is likely to diffuse through it. Consider for example a new science fiction book one would like to market by word-of-mouth. If science fiction fans are scattered throughout the network, with very few recommendations shared between them, then recommendations about the new book are unlikely to diffuse. If on the other hand one finds one or more science fiction *communities*, where sci-fi fans are close together in the network because they exchange recommendations with one another, then the book recommendation has a chance of spreading by word-of-mouth.

In the following analysis, we use a community finding algorithm [Clauset et al., 2004] in order to discover the types of products that link customers and so define a community. The algorithm breaks up the component into parts, such that the modularity Q ,

$$Q = (\text{number of edges within communities}) - (\text{expected number of such edges}), \quad (6.6)$$

is maximized. In other words, the algorithm identifies communities such that individuals within those communities tend to preferentially exchange recommendations with one another.

The results of the community finding analysis, while primarily descriptive, illustrate both the presence of communities whose members are linked by their common interests, and the presence cross-cutting interests between communities. Applying the algorithm to the largest component, we identify many small communities and a few larger ones. The largest contains 21,000 nodes, 5,000 of whom are senders of a relatively modest 335,000 recommendations. More interesting than simply observing the size of communities is discovering what interests bring them together. We identify those interests by observing product categories where the number of recommendations within the community is significantly higher than it is for the overall customer population. Let p_c be the proportion of all recommendations that fall within a particular product category c . Then for a set of individuals sending x_g recommendations, we would expect by chance that $x_g * p_c \pm \sqrt{x_g * p_c * (1 - p_c)}$ would fall within category c . We note the product categories for which the observed number of recommendations in the community is many standard deviations higher than expected. For example, compared to the background population, the largest community is focused on a wide variety of books and music. In contrast, the second largest community, involving 10,412 individuals (4,205 of whom are sending over 3 million recommendations), is predominantly focused on DVDs from many different genres, with no particular emphasis on anime. The anime community itself emerges as a highly unusual group of 1,874 users who exchanged over 3 million recommendations.

We find that large communities are very diverse and uninteresting. Perhaps the most interesting are the medium sized communities, some of which are listed in Table 6.6, having around 100 senders and often reflecting specific interests. Among the hundred or so medium communities, we found, for example, several communities focusing on Christianity. While some of the Christian communities also shared an interest in children's books, Broadway musicals, and travel to Italy, others focused on prayer and bibles, still others also enjoyed DVDs of the Simpsons TV series, and others still took an interest in Catholicism, occult spirituality and kabbalah.

Communities were usually centered around a product group, such as books, music, or DVDs, but almost all of them shared recommendations for all types of products. The DVD communities ranged from bargain shoppers purchasing discounted comedy and action DVDs to smaller anime or independent movie communities, to a group of customers purchasing predominantly children's movies. One community focused heavily on indie music, and imported dance and club music. Another seemed to center around intellectual pursuits, including reading books on sociology, politics, artificial intelligence, mathematics, and media

# nodes	# senders	topics
735	74	books: American literature, poetry
710	179	sci-fi books, TV series DVDs, alternative rock music
667	181	music: dance, indie
653	121	discounted DVDs
541	112	books: art & photography, web development, graphical design, sci-fi
502	104	books: sci-fi and other
388	77	books: Christianity and Catholicism
309	81	books: business and investing, computers, Harry Potter
192	30	books: parenting, women's health, pregnancy
163	48	books: comparative religion, Egypt's history, new age, role playing games

Table 6.6: A sample of the medium sized communities present in the largest component

culture, listening to classical music and watching neo-noir film. Several communities centered around business and investment books and frequently also recommended books on computing. One business and investment community included fans of the Harry Potter fiction series, while another enjoyed science fiction and adventure DVDs. One of communities with the most particular interests recommended not only business and investing books to one another, but also an unusual number of books on terrorism, bacteriology, and military history. A community of what one can presume are web designers recommended books to one another on art and photography, web development, graphical design, and Ray Bradbury's science fiction novels. Several sci-fi TV series such as Buffy the Vampire Slayer and Star Trek appeared prominently in a few communities, while Stephen King and Douglas Clegg featured in a community recommending horror, sci-fi, and thrillers to one another. One community focused predominantly on parenting, women's health and pregnancy, while another recommended a variety of books but especially a collection of cookie baking recipes.

Going back to components in the network that were disconnected from the largest component, we find similar patterns of homophily, the tendency of like to associate with like. Two of the components recommended technical books about medicine, one focused on dance music, while some others predominantly purchased books on business and investing. Given more time, it is quite possible that one of the customers in one of these disconnected components would have received a recommendation from a customer within the largest component, and the two components would have merged. For example, a disconnected component of medical students purchasing medical textbooks might have sent or received a recommendation from the medical community within the largest component. However, the medical community may also become linked to other parts of the network through a different interest of one of its members. At the very least many communities, no matter their focus, will have recommendations for children's books or movies, since children are a focus for a great many people. The community finding algorithm on the other hand is able to break up the larger social network to automatically identify groups of individuals with a particular focus or a set of related interests. Now that we have shown that communities of customers recommend types of products reflecting their interests, we will examine whether these different kinds of products tend to have different success rates in their recommendations.

6.7.2 Recommendation effectiveness by book category

Some contexts result in social ties that are more effective at conducting an action. For example, in small world experiments, where participants attempt to reach a target individual through their chain of acquaintances, profession trumped geography, which in turn was more useful in locating a target than attributes such as religion or hobbies [Killworth and Bernard, 1978, Travers and Milgram, 1969]. In the context of product recommendations, we can ask whether a recommendation for a work of fiction, which may be made by any friend or neighbor, is more or less influential than a recommendation for a technical book, which may be made by a colleague at work or school.

Table 6.7 shows recommendation trends for all top level book categories by subject. For clarity, we group the results by 4 different category types: fiction, personal/leisure, professional/technical, and non-fiction/other. Fiction encompasses categories such as Sci-Fi and Romance, as well as children's and young adult books. Personal/Leisure encompasses everything from gardening, photography and cooking to health and religion.

First, we compare the relative number of recommendations to reviews posted on the site (column c_{av}/r_{p1} of table 6.7). Surprisingly, we find that the number of people making personal recommendations was only a few times greater than the number of people posting a public review on the website. We observe that fiction books have relatively few recommendations compared to the number of reviews, while professional and technical books have more recommendations than reviews. This could reflect several factors. One is that people feel more confident reviewing fiction than technical books. Another is that they hesitate to recommend a work of fiction before reading it themselves, since the recommendation must be made at the point of purchase. Yet another explanation is that the median price of a work of fiction is lower than that of a technical book. This means that the discount received for successfully recommending a mystery novel or thriller is lower and hence people have less incentive to send recommendations.

Next, we measure the per category efficacy of recommendations by observing the ratio of the number of purchases occurring within a week following a recommendation to the number of recommenders for each book subject category (column b_r of table 6.7). On average, only 2% of the recommenders of a book received a discount because their recommendation was accepted, and another 1% made a recommendation that resulted in a purchase, but not a discount. We observe marked differences in the response to recommendation for different categories of books. Fiction in general is not very effectively recommended, with only around 2% of recommenders succeeding. The efficacy was a bit higher (around 3%) for non-fiction books dealing with personal and leisure pursuits. Perhaps people generally know what their friends' leisure interests are, or even have gotten to know them through those shared interests. On the other hand they may not know as much about each others' tastes in fiction. Recommendation success is highest in the professional and technical category. Medical books have nearly double the average rate of recommendation acceptance. This could be in part attributed to the higher median price of medical books and technical books in general. As we will see in Section 6.8.2, a higher product price increases the chance that a recommendation will be accepted.

Recommendations are also more likely to be accepted for certain religious categories: 4.3% for Christian living and theology and 4.8% for Bibles. In contrast, books not tied to organized religions, such as ones on the subject of new age (2.5%) and occult (2.2%) spirituality, have lower recommendation effectiveness. These results raise the interesting possibility that individuals have greater influence over one another in an organized context, for example through a professional contact or a religious one. There are exceptions of course. For example, Japanese anime DVDs have a strong following in the US, and this is reflected

category	n_p	N	cc	r_{p1}	v_{av}	c_{av}/r_{p1}	p_m	$b_r * 100$
Books general	370,230	2,860,714	1.87	5.28	4.32	1.41	14.95	3.12
Fiction								
Children	46,451	390,283	2.82	6.44	4.52	1.12	8.76	2.06**
Literature	41,682	502,179	3.06	13.09	4.30	0.57	11.87	2.82*
Mystery	10,734	123,392	6.03	20.14	4.08	0.36	9.60	2.40**
Science fiction	10,008	175,168	6.17	19.90	4.15	0.64	10.39	2.34**
Romance	6,317	60,902	5.65	12.81	4.17	0.52	6.99	1.78**
Teens	5,857	81,260	5.72	20.52	4.36	0.41	9.56	1.94**
Comics	3,565	46,564	11.70	4.76	4.36	2.03	10.47	2.30*
Horror	2,773	48,321	9.35	21.26	4.16	0.44	9.60	1.81**
Personal								
Religion	43,423	441,263	1.89	3.87	4.45	1.73	9.99	3.13
Health/Body	33,751	572,704	1.54	4.34	4.41	2.39	13.96	3.04
History	28,458	28,3406	2.74	4.34	4.30	1.27	18.00	2.84
Home/Garden	19,024	180,009	2.91	1.78	4.31	3.48	15.37	2.26**
Entertainment	18,724	258,142	3.65	3.48	4.29	2.26	13.97	2.66*
Arts/Photo	17,153	179,074	3.49	1.56	4.42	3.85	20.95	2.87
Travel	12,670	113,939	3.91	2.74	4.26	1.87	13.27	2.39**
Sports	10,183	120,103	1.74	3.36	4.34	1.99	13.97	2.26**
Parenting	8,324	182,792	0.73	4.71	4.42	2.57	11.87	2.81
Cooking	7,655	146,522	3.02	3.14	4.45	3.49	13.97	2.38*
Outdoors	6,413	59,764	2.23	1.93	4.42	2.50	15.00	3.05
Professional								
Professional	41,794	459,889	1.72	1.91	4.30	3.22	32.50	4.54**
Business	29,002	476,542	1.55	3.61	4.22	2.94	20.99	3.62**
Science	25,697	271,391	2.64	2.41	4.30	2.42	28.00	3.90**
Computers	18,941	375,712	2.22	4.51	3.98	3.10	34.95	3.61**
Medicine	16,047	175,520	1.08	1.41	4.40	4.19	39.95	5.68**
Engineering	10,312	107,255	1.30	1.43	4.14	3.85	59.95	4.10**
Law	5,176	53,182	2.64	1.89	4.25	2.67	24.95	3.66*
Other								
Nonfiction	55,868	560,552	2.03	3.13	4.29	1.89	18.95	3.28**
Reference	26,834	371,959	1.94	2.49	4.19	3.04	17.47	3.21
Biographies	18,233	277,356	2.80	7.65	4.34	0.90	14.00	2.96

Table 6.7: Statistics by book category: n_p : number of products in category, N : number of customers, cc : percentage of customers in the largest connected component, r_{p1} : avg. # reviews in 2001 – 2003, v_{av} : average star rating, c_{av} : average number of people recommending product, c_{av}/r_{p1} : ratio of recommenders to reviewers, p_m : median price, b_r : ratio of the number of purchases resulting from a recommendation to the number of recommenders. The symbol ** denotes statistical significance at the 0.01 level, * at the 0.05 level.

in their frequency and success in recommendations. Another example is that of gardening. In general, recommendations for books relating to gardening have only a modest chance of being accepted, which agrees with the individual prerogative that accompanies this hobby. At the same time, orchid cultivation can be a highly organized and social activity, with frequent ‘shows’ and online communities devoted entirely to orchids. Perhaps because of this, the rate of acceptance of orchid book recommendations is twice as high as those for books on vegetable or tomato growing.

6.8 Products and recommendations

We have examined the properties of the recommendation network in relation to viral marketing. Now we focus on the products themselves and their characteristics that determine the success of recommendations.

6.8.1 How long is the long tail?

Recently a ‘long tail’ phenomenon has been observed, where a large fraction of purchases are of relatively obscure items where each of them sells in very low numbers but there are many of those items. On Amazon.com, somewhere between 20 to 40 percent of unit sales fall outside of its top 100,000 ranked products [Brynjolfsson et al., 2003]. Considering that a typical brick and mortar store holds around 100,000 books, this presents a significant share. A streaming-music service streams more tracks outside than inside its top 10,000 tunes [Anonymous, 2005].

We performed a similar experiment using our data. Since we do not have direct sales data we used the number of successful recommendations as a proxy to the number of purchases. Figure 6.15 plots the distribution of the number of purchases and the number of recommendations per product. Notice that both the number of recommendations and the number of purchases per product follow a heavy-tailed distribution and that the distribution of recommendations has a heavier tail.

Interestingly, figure 6.15(a) shows that just the top 100 products account for 11.4% of the all sales (purchases with discount), and the top 1000 products amount to 27% of total sales through the recommendation system. On the other hand 67% of the products have only a single purchase and they account for 30% of all sales. This shows that a significant portion of sales come from products that sell very few times. Recently there has been some debate about the long tail [Gomes, 2006, Anderson, 2006]. Some argue that the presence of the long tail indicates that niche products with low sales are contributing significantly to overall sales online. We also find that the tail is a bit longer than the usual 80-20 rule, with the top 20% of the products contributing to about half the sales. It is important to note, however, that our observations do not reflect the total sales of the products on the website, since they include only successful recommendations that resulted in a discount. This incorporates both a bias in the kind of product that is likely to be recommended, and in the probability that a recommendation for that kind of product is accepted.

If we look at the distribution in the number of recommendations per product, shown in Figure 6.15(b), we observe an even more skewed distribution. 30% of the products have only a single recommendation and the top 56,000 most recommended products (top 10%) account for 84% of all recommendations. This is consistent with our previous observations some types of products, *e.g.* anime DVDs, are more heavily recommended than others.

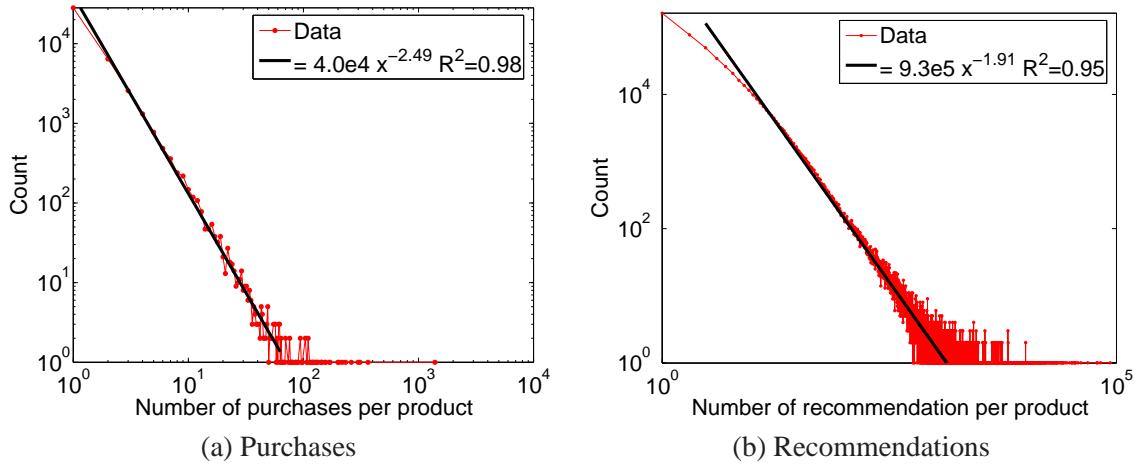


Figure 6.15: Distribution of number of purchases and recommendations of a product. (a) shows the number of purchases that resulted in a discount per product, and (b) shows the distribution of the number of recommendations per product.

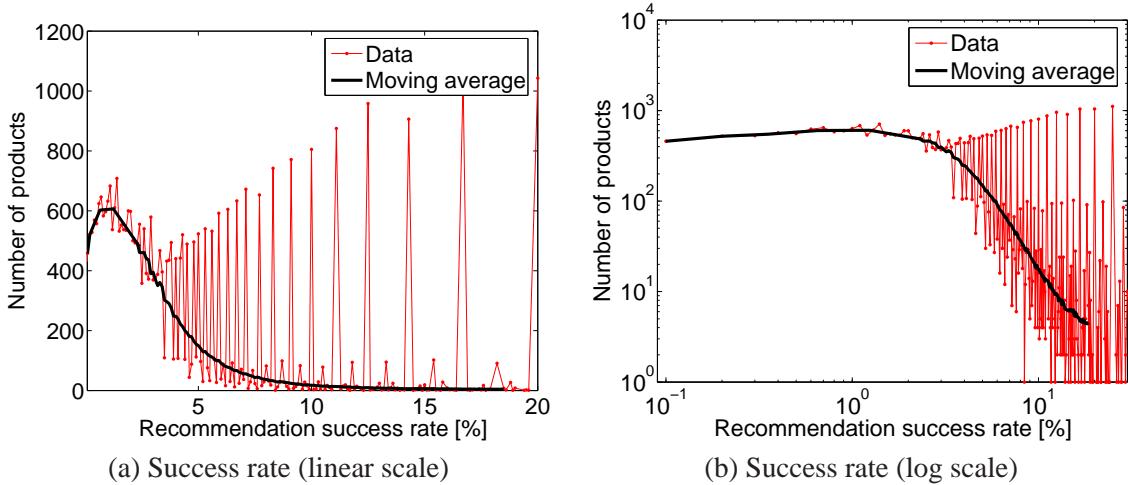


Figure 6.16: Distribution of product recommendation success rates. Both plots show the same data: (a) on a linear (lin-lin) scale, and (b) on a logarithmic (log-log) scale. The bold line presents the moving average smoothing.

Next we examine the distribution of the product recommendation success rate. Out of more than half a million products we took all the products with at least a single purchase, of which there are 41,000 (7%). Figure 6.16 shows the success rate (purchases/recommendations). Notice that the distribution is not heavy tailed and has a mode at around 1.3% recommendation success rate. 55% of the products have a success rate bellow 5% and there are around 14% of the products that have a recommendation success rate higher than 20%.

6.8.2 Modeling the product recommendation success

So far we have seen that some products generate many recommendations and some have a better return than others on those recommendations, but one question still remains: what determines the prod-

uct's viral marketing success? We present a model which characterizes product categories for which recommendations are more likely to be accepted. We use a regression of the following product attributes to correlate them with recommendation success:

- N : number of nodes in the social network (number of unique senders and receivers)
- N_s : number of senders of recommendations
- N_r : number of recipients of recommendations
- r : number of recommendations
- E : number of edges in the social network (number of unique (sender, receiver) pairs)
- p : price of the product
- v : number of reviews of the product
- t : average product rating

From the original set of the half-million products, we compute a success rate s for the 8,192 DVDs and 50,631 books that had at least 10 recommendation senders and for which a price was given. In section 6.7.2 we defined recommendation success rate s as the ratio of the total number purchases made through recommendations and the number of senders of the recommendations. We decided to use this kind of normalization, rather than normalizing by the total number of recommendations sent, in order not to penalize communities where a few individuals send out many recommendations (figure 6.3(b)). Note that in general s could be greater than 1, but in practice this happens extremely rarely (there are only 107 products where $s > 1$ which were discarded for the purposes of this analysis).

Since the variables follow a heavy tailed distribution, we use the following model:

$$s = \exp\left(\sum_i \beta_i \ln(x_i) + \epsilon_i\right) \quad (6.7)$$

where x_i are the product attributes (as described on previous page), and ϵ_i is random error.

We fit the model using least squares and obtain the coefficients β_i shown in table 6.9. With the exception of the average rating, they are all significant, but just the number of recommendations alone accounts for 15% of the variance (taking all eight variables into consideration yields an R2 of 0.30 for books and 0.81 for DVDs). We should also note that the variables in our model are highly collinear, as can be seen from the pairwise correlation matrix (table 6.8). For example, the number of recommendations r is highly negatively correlated with the dependent variable ($\ln(s)$) but in the regression model it exhibits positive influence on the dependent variable. This is probably due to the fact that the number of recommendations is naturally dependent on the number of senders and number of recipients, but it is the high number of recommendations relative to the number of senders that is of importance.

To illustrate the dependencies between the variables we train a Bayesian dependency network [Chickering, 2003], and show the learned structure for the combined (Books and DVDs) data in figure 6.17. In this a directed acyclic graph where nodes are variables, and directed edges indicate that the distribution of a child depends on the values taken in the parent variables.

Notice that the average rating (t) is not predictive of the recommendation success rate (s). It is no surprise that the number of recommendations r is predictive of number of senders n_s . Similarly, the number of

	$\ln(s)$	$\ln(N)$	$\ln(N_s)$	$\ln(N_r)$	$\ln(r)$	$\ln(E)$	$\ln(p)$	$\ln(v)$	$\ln(t)$
$\ln(s)$	1								
$\ln(N)$	0.275	1							
$\ln(N_s)$	0.103	0.907	1						
$\ln(N_r)$	0.310	0.994	0.864	1.000					
$\ln(r)$	0.396	0.979	0.828	0.988	1				
$\ln(E)$	0.392	0.981	0.831	0.990	0.999	1			
$\ln(p)$	0.185	0.098	0.088	0.098	0.107	0.106	1		
$\ln(v)$	-0.050	0.465	0.490	0.449	0.421	0.423	-0.053	1	
$\ln(t)$	-0.031	0.064	0.071	0.061	0.056	0.056	-0.019	0.269	1

Table 6.8: Pairwise Correlation Matrix of the Books and DVD Product Attributes. $\ln(s)$: log recommendation success rate, $\ln(N)$: log number of nodes, $\ln(N_s)$: log number of senders of recommendations, $\ln(N_r)$: log number of receivers, $\ln(r)$: log number of recommendations, $\ln(E)$: log number of edges, $\ln(p)$: log price, $\ln(v)$: log number of reviews, $\ln(t)$: log average rating.

Variable	Books	DVD
	Coefficient β_i	Coefficient β_i
const	1.317 (0.0038) **	0.929 (0.0100) **
N	-0.579 (0.0060) **	0.171 (0.0124) **
N_s	0.144 (0.0018) **	-0.070 (0.0023) **
N_r	-0.006 (0.0064)	-0.360 (0.0104) **
r	0.062 (0.0084) **	-0.002 (0.0083)
E	0.383 (0.0106) **	0.251 (0.0088) **
p	0.013 (0.0003) **	0.007 (0.0016) **
v	-0.003 (0.0001) **	-0.003 (0.0006) **
t	-0.001 (0.0006) *	0.000 (0.0009)
R^2	0.30	0.81

Table 6.9: Regression Using the Log of the Recommendation Success Rate $\log(s)$, as the Dependent Variable for Books and DVDs separately. For each coefficient we provide the standard error and the statistical significance level (**:0.001, *:0.1). We fit separate models for books and DVDs.

edges e is predictive of number of senders n_s . Interestingly, price p is only related to the number of reviews v . Number of recommendations r , number of senders n_s and price p , are directly predictive of the recommendation success rate s .

Returning to our regression model, we find that the numbers of nodes and receivers have negative coefficients, showing that successfully recommended products are actually more likely to be not so widely popular. The only attributes with positive coefficients are the number of recommendations r , number of edges e , and price p . This shows that more expensive and more recommended products have a higher success rate. These recommendations should occur between a small number of senders and receivers, which suggests a very dense recommendation network where lots of recommendations are exchanged between a small community of people. These insights could be of use to marketers — personal recommendations are most effective in small, densely connected communities enjoying expensive products.

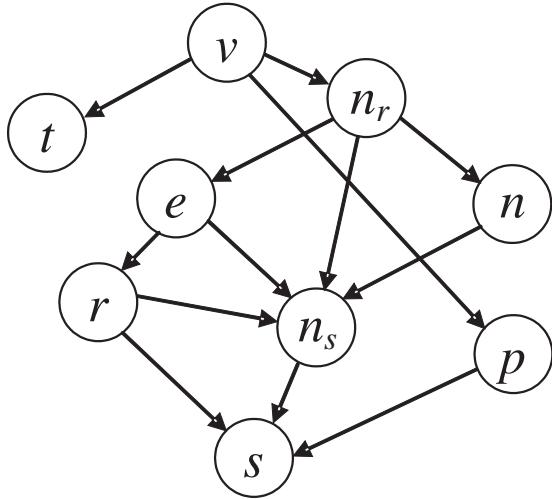


Figure 6.17: A Bayesian network showing the dependencies between the variables. s : recommendation success rate, n : number of nodes, n_s : number of senders of recommendations, n_r : log number of receivers, r : number of recommendations, e : number of edges, p : price, v : number of reviews, t : average rating.

6.9 Cascade shapes in viral marketing

Information cascades are phenomena whereby individuals adopt a new action or idea due to influence by others. As such a process spreads through an underlying social network, it can result in widespread adoption overall. We consider information cascades in the context of recommendations, and in particular study the patterns of cascading recommendations that arise in large social networks. We investigate a large person-to-person recommendation network, consisting of four million people who made sixteen million recommendations on half a million products. Such a dataset allows us to pose a number of fundamental questions: What cascades arise frequently in real life? What features distinguish them? We enumerate and count cascade subgraphs on large directed graphs; as one component of this, we develop a novel efficient heuristic based on graph isomorphism testing that scales to large datasets. We discover novel patterns: the distribution of cascade sizes and depths follows a power law. Generally, cascades tend to be shallow, but occasional large bursts of propagation can occur. Cascade subgraphs are mainly tree-like, but we observe variability in connectivity and branching across recommendations for different types of products.

6.9.1 Cascades

The social network of interactions between a group of individuals plays a fundamental role in the spread of information, ideas, innovation, and influence among its members. The network effect has been observed in many cases, where an idea or action gains sudden widespread popularity through word of mouth or viral marketing. For example, some movies become widely popular through word-of-mouth advertising. Google's Gmail service captured a significant market share in spite of the fact that up to recently the *only* way to obtain a free email account is through a referral. One can also find many examples in weblogs (blogs), where a story or piece of information gets widely referred to by the blogger community and is eventually picked up by the mass media.

Information cascades are phenomena where an action or idea becomes widely adopted due to influence by others, as opposed to individual reasoning in isolation [Bikhchandani et al., 1992]. Cascades are also known as “fads” or “resonance.” There has been significant work done in modeling the spread and adoption of ideas and influence through a social network [Goldenberg et al., 2001, Granovetter, 1978, Domingos and Richardson, 2001, Kempe et al., 2003, Richardson and Domingos, 2002b].

The formalism for cascades is activation of nodes in a graph where nodes represent individuals, edges relationships, and a binary node state shows whether a person is part of the cascade. The chance that a node is activated is influenced by the state of its neighbors. A related formalism is a graph where the nodes are agents and a directed edge (i, j, t) indicates that i influenced j at time t .

Cascades have been studied for many years by sociologists concerned with the *diffusion of innovation* [Rogers, 1995]; more recently, researchers have investigated cascades for the purpose of selecting trendsetters [Domingos and Richardson, 2001, Richardson and Domingos, 2002b], finding inoculation targets in epidemiology [Newman, 2002], and explaining trends in blogosphere [Adar and Adamic, 2005, Adar et al., 2004, Gruhl et al., 2004, Kumar et al., 2003]. To our knowledge, however, the difficulty in obtaining data has limited the extent of analysis on large-scale, complete datasets representing cascades. Here we look at the patterns of influence in a large-scale, real recommendation network and examine the topological structure of cascades.

Here we ask the question: What cascades arise frequently in real life? Are they like trees, stars, or something else? We describe a large person-to-person recommendation network, consisting of 4 million people who made 16 million recommendations on half a million products in section 6.9.2. To analyze the data, we first create graphs where incoming edges influenced the creation of outgoing edges. We remove edges that violate the temporal requirement of a cascade (*i.e.*, influence must be exerted before the effect). Then, we enumerate and count all possible cascade subgraphs using an algorithm developed in section 6.9.3. Therein, we propose a heuristic for graph isomorphism involving the degree distribution and the eigenvalues of the adjacency matrix that scales to large datasets. We apply the algorithm to the recommendation dataset, and analyze it in section 6.9.4.

We find novel patterns and the analysis of the results gives us insight into the cascade formation process. We find that distribution of sizes and depths of cascades follows a heavy-tailed distribution. Generally cascades are shallow but occasional large bursts also occur. The cascade sub-patterns reveal mostly small tree-like subgraphs; however we observe differences in connectivity and the shape of cascades across product groups. We find common cases when people who do not link to each other recommend to the same set of friends; and cases where recommendation propagates but then returns to the same people.

6.9.2 The recommendation network

We study a recommendation network dataset from a large on-line retailer we described earlier in the chapter. In brief, the recommendation network consists of 15,646,121 recommendations made among 3,943,084 distinct users from June 2001 to May 2003 (711 days). A total of 542,719 different products belonging to four product categories (Books, DVDs, Music and Videos) were recommended.

We represent this relational dataset as a directed multigraph: nodes represent customers, and a directed edge (i, j, p, t) means that node i recommended product p to customer j at time t . The typical edge generation process is as follows: a node (person) i first buys product p at time t , and then recommends it to nodes $\{j_1, \dots, j_n\}$. The j nodes can then buy the product (with the option to recommend it to others).

Note that even if all nodes j buy the product, only the first purchaser will get the discount, which is marked by a purchase flag (*buy-bit*). We cannot directly use the buy-bit to determine whether a recommendation caused a purchase. In addition to the buy-bit, we also record the number of customers who recommended the product (since they had to buy the product to recommend it). We extract per-group recommendation networks by taking the edge-induced subgraph formed by all the products of a given category. Table 6.2 gives the basic network statistics and observations.

6.9.3 Proposed method

In this section we present the algorithms and techniques developed to efficiently enumerate and count frequent recommendation patterns in a large graph, including a heuristic for subgraph isomorphism.

Ideally one would expect cascades to be trees or near-trees. We soon found out that recommendations create arbitrary graphs: there are multiple recommendations on the same product or multiple products between the nodes, there are multiple purchases of the same product, and one finds many cycles.

To find cascades one first needs to identify cases when incoming recommendations could cause purchases and further outgoing recommendations. Recommendations into node u that precede a purchase can be posited to have influenced the purchase. There are two ways to establish this. If an edge is marked by a purchase flag, we assume the recommendation influenced the purchase. Alternately, the existence of two directed edges (i, j, p, t) and (j, k, p, t') for $t' > t$ suggests cascade behavior. That is, node j receives a recommendation for product p at time t and then makes recommendation for the same product at a later time t' .

First we create a separate graph of recommendations for each product. To find cascades we propose the following two-step procedure:

Delete late recommendations:

To keep only recommendations that influenced the purchase we *delete late recommendations*: given a single product recommendation network, for every node we delete all incoming recommendations (edges) that happened after the first purchase of a product. This procedure removes all recommendations of the product a person received after the first purchase. This guarantees that for every node the time of all incoming edges is strictly smaller than the time of all outgoing edges.

Delete no-purchase nodes:

Preliminary data analysis showed that the majority of recommendations do not produce cascades. We also observed many star-like patterns where the center node recommends to a large number of people, none of whom purchase the product. This occurs frequently in DVD subgraphs. To prevent this type of large but shallow pattern, we delete all nodes that did not purchase the product.

After deleting late recommendations each connected component corresponds to a cascade. All paths in the component are time-increasing (*i.e.*, a cascade subgraph contains only directed paths with strictly increasing edge times). Deleting no-purchase nodes ensures that we detect only true cascade patterns.

Cascade enumeration:

Next we enumerate all possible cascades. In preliminary experiments we first enumerated the maximal cascades, which after the steps described above reduces down to enumerating all connected components of the network. This approach works well and is very fast, but suffers from the fact that the counts are small. Here we take a different approach. Since we are interested in purely topological properties of the cascades, rather than enumerating all possible connected subgraphs, we enumerate all *local cascades*. This means that for every node we explore the cascade in the neighborhood around the node. For every node n , we create a graph induced on nodes up to H hops away from n , where H ranges from 1 up to the distance to the farthest node. One can think of this as exploring node n 's neighborhood 1, 2, 3,... steps away. This way for every node we capture the local structure of the cascade around it at various distances.

Approximate graph isomorphism:

An essential step in counting cascades is determining whether a new cascade is isomorphic to a previously discovered graph. No polynomial-time algorithm is known for the graph isomorphism problem, and so we resort to an approximate, heuristic solution. For each graph we create a *signature*. A good signature is one where isomorphic graphs have the same signature, but where few non-isomorphic graphs share the same signature.

We propose a multi-level approach where the computational complexity (and accuracy) of the graph isomorphism resolution depends on the size of the graph. For smaller graphs we perform an exact isomorphism test; as the size of the graph increases this becomes prohibitively expensive so we use gradually simpler but faster techniques which give only approximate solutions. Another trick is that for each graph we create an efficiently computable signature, use hashing, and then use more expensive isomorphism tests only on graphs with the same signature.

For every graph we create a signature which is composed of the number of nodes, the number of edges, and the sorted in- and out-degree sequence. For graphs with fewer than 500 nodes, we also include the singular values of the adjacency matrix (via singular value decomposition). As singular values are continuous we round them to 4 most significant digits and then hash the values.

We then hash the graphs using the signatures. Additionally, for graphs with fewer than 9 nodes we perform exact isomorphism checking. When the isomorphism check is used, we keep a list of all variants of graphs that collided. Since we first hash signatures, we then check for isomorphism only the graphs with the same signature. So the number of true isomorphism checks is very small.

Note that a small minority of cascades are larger than 9 nodes, so for most of the subgraphs we get the exact solution; as the cascade size increases the number of occurrences decreases, and this is where we make use of an approximate solution.

We performed a small set of experiments to evaluate the proposed approximate graph isomorphism algorithm. Given a graph with 8 nodes and 12 edges 100,000 brute-force evaluations of graph-isomorphism took under 40 seconds on a standard desktop. In the second experiment we generated 100,000 random graphs (Erdős-Rényi model), each of them with a randomly chosen number of nodes between 4 to 20 and twice as many edges (average degree of 2). The counting took 50 seconds. In this experiment we observed at most 53 non-isomorphic graphs (5 nodes, 10 edges) with the same signature. At the end the random generation created a total of 6,194 5-node graphs, of which 1,601 were non-isomorphic.

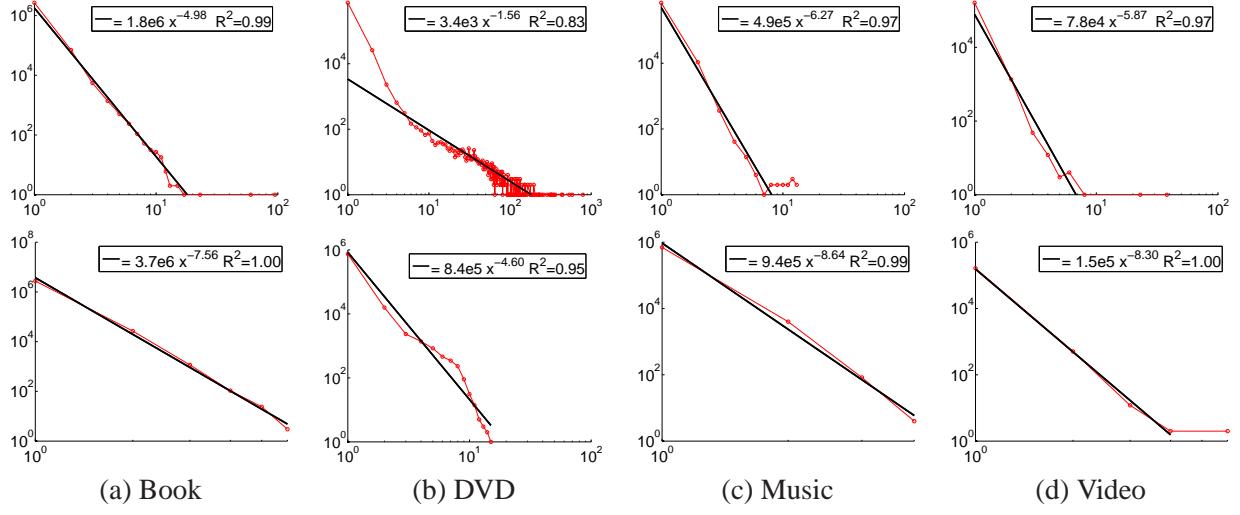


Figure 6.18: Size and depth distribution of the cascades for the four product groups. Top row shows the size distribution of the cascades (log size of cascade vs. log count). Bottom row shows the distribution of the depths of the cascades (log depth of the cascade vs. log count). Bold line presents a power-law fit.

This analysis shows that we are able to efficiently find and enumerate cascades even in a large recommendation network. The graph isomorphism checking is fast and scalable to serve our purpose.

6.9.4 Patterns of recommendation

Size and depth distribution of cascades

We measure the size of the cascade in terms of the number of nodes and the depth, which is the length of the longest directed path in the cascade. As in all experiments we create per-product recommendation networks, delete late recommendations and no-purchase nodes, and then perform the analysis.

Figure 6.18 shows the distribution of cascade sizes (top row) and depths (bottom row) for the four product groups. The size of cascades follows a power law. For books the largest cascade has 95 nodes and 231 edges. For DVDs the largest cascade is eight times larger ($N = 791, E = 5544$). The cascades involving music or videos are much smaller, the largest cascades are $N = 13, E = 56$ and $N = 37, E = 169$ respectively.

The slopes of the power-fits (top row of figure 6.18) reveal that DVDs had the highest proportion of large cascades, as its power coefficient is the largest. For music the fraction of large cascades is much smaller. While the first part of the size distribution for DVDs (figure 6.18(b)) has slope -4.5 , which is close to the other three product groups, the curve then flattens to -1.5 .

The depth distribution, figure 6.18, shows that cascades are generally shallow except for DVDs. The maximum depth of a cascade is 6 for books, 15 for DVDs, 4 for music, and 6 for videos. So DVDs have the strongest evidence for cascades.

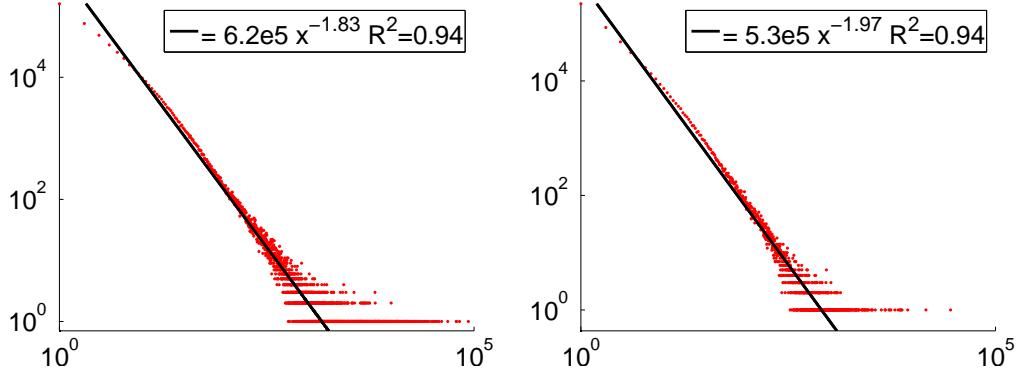


Figure 6.19: Distribution of recommendations and purchases over the products: number of recommendations of the product vs. count (left); number of purchases vs. count (right).

One might posit that cascades are branching processes. However it is known that for a particular run of a branching process, the distribution of depths, conditioned on the size being finite, is exponential. In other words, if cascades were purely branching processes, then the depths should be exponentially distributed. Figure 6.18 shows that the depth distribution follows a power law; that is, we are observing more of deep cascades than expected under a branching process.

There are a number of possible explanations for this phenomena: cascades can collide, increasing the probability of success in some part of the social network [Leskovec et al., 2006a, 2007a]. Cascade sizes also reflect an underlying power law in sales frequencies, as shown in figure 6.19. The number of purchases decays faster than the recommendations. And in the stochastic cascade generation process we proposed here the cascade size distribution follows a power law with the exponent -1 .

Frequent cascade subgraphs

What cascades arise frequently in real life? Are they like trees, stars, long chains, or something else? We now explore the building blocks of the cascades, by performing the following procedure. For each product recommendation graph, we first identify cascades (delete late recommendations and no-purchase nodes). Then for each node we create a subgraph on nodes at distance at most h hops, where h varies from 1 up to the value where all nodes in the cascade are reached. We then count the graphs using the approximate graph isomorphism technique described in section 6.9.3.

General observations: For books we identified a total of 122,657 cascades, of which 959 are topologically different. There are 213 cascades that occur at least ten times. For DVDs we identified 289,055 cascades, of which 87,614 are topologically different. There are 3,015 cascades that occur at least ten times. For music we identified 13,330 cascades, of which 158 were topologically different. Only 23 cascades occurred at least ten times. Videos contained the least evidence for cascades, with 1,928 subgraphs containing 109 unique patterns. Only 12 subgraphs occurred more than ten times.

The number of cascades concur with observations made from figure 6.18 and table 6.2, where DVDs had the largest and richest set of cascades. Since DVDs contain the deepest cascade, there is more opportunity for topological variety than on the other products types. Even though the music network is three times larger than the video network, it does not exhibit much larger topological variety.

Id	Graph	Nodes	Edges	Book		DVD		Music		Video	
				R	F	R	F	R	F	R	F
G_1		2	1	1	86,430	1	36,863	1	11,518	1	1,425
G_2		3	2	2	10,573	4	3,238	2	492	5	33
G_3		3	2	3	5,089	2	5,147	3	389	3	61
G_4		3	2	6	1,593	5	2419	5	115	22	4
G_5		3	3	4	3,115	3	4746	4	201	2	63
G_6		4	3	5	2,769	15	505	6	55	20	5
G_7		4	3	8	726	25	416	7	30	27	4
G_8		4	3	10	598	7	909	8	25	0	0
G_9		4	3	12	398	33	312	13	12	0	0
G_{10}		4	3	13	362	22	424	9	18	26	4
G_{11}		4	3	18	156	37	276	53	4	0	0
G_{12}		4	3	29	82	24	418	28	8	0	0
G_{13}		4	3	92	21	12	549	54	4	0	0
G_{14}		4	4	9	625	11	552	31	7	13	8
G_{15}		4	4	22	112	16	495	10	15	0	0
G_{16}		4	4	23	111	20	435	57	3	0	0
G_{17}		4	4	26	85	17	485	83	2	0	0
G_{18}		4	4	30	79	9	706	32	7	29	3
G_{19}		4	4	37	64	38	273	24	9	0	0
G_{20}		4	4	47	51	955	28	0	0	0	0
G_{21}		4	4	90	21	857	31	0	0	0	0
G_{22}		4	4	91	21	1368	20	0	0	0	0

Table 6.10: Frequent cascades for the 4 product groups. We show all graphs up to 4 nodes and 4 edges. Ordered by size. For each graph we show rank (R) and frequency (F).

Analysis of frequent cascade patterns: Table 6.10 shows ranks R and frequencies F of 22 cascades for the 4 product groups. Cascades are ordered by size. The table also includes all sub-cascades with at most four nodes and four edges. Interesting, 14 cascade patterns can be observed in all the product groups. Table 6.10 shows ten of them.

The most common cascade, G_1 , represents a single recommendation. This pattern accounts for 70% of all book cascades, 86.4% of all music cascades, 74% of all video cascades, but just 12.8% of DVD cascades. The chain of three nodes (G_3) is the most common depth two cascade, accounting for 4.1% of book cascades, about 3% of video and music cascades, but only 1.8% of DVD cascades. DVD cascades tend to be most densely linked.

Comparing G_2 and G_4 shows that simple splits are more frequent than collisions. For books there are

Id	Graph	Nodes	Edges	Book		DVD		Music		Video	
				R	F	R	F	R	F	R	F
G_{23}		4	5	14	274	23	422	0	0	0	0
G_{24}		4	5	34	77	75	171	38	5	28	3
G_{25}		4	5	84	23	52	216	0	0	109	1
G_{26}		4	6	24	105	6	1299	27	8	6	29
G_{27}		5	4	7	1024	74	174	20	10	0	0
G_{28}		5	4	16	211	332	62	47	5	0	0
G_{29}		5	4	50	47	333	62	64	3	0	0
G_{30}		5	4	53	41	282	69	48	5	0	0
G_{31}		5	4	60	31	1045	26	158	1	0	0
G_{32}		5	4	72	27	822	32	21	10	0	0
G_{34}		5	9	137	14	131	119	55	3	15	7
G_{35}		5	10	125	15	18	452	155	1	10	16

Table 6.11: Some larger frequent cascades for 4 product groups. Ordered by size. For each graph we show rank (R) and frequency (F).

6.6 times more splits than collisions; for DVDs this factor drops to 1.3; and it is 4.2 and 8.25 for music and videos respectively. Very similar observations hold for splits and collisions on 4 nodes (G_6 and G_{13}); however notice that for DVDs the collision of 3 nodes (G_{13}) is slightly more frequent than the split (G_6). Another such example of reversed graphs are G_7 , G_{11} and G_8 , G_{12} . Again, the split pattern is more frequent than the collision. The ratio is more unbalanced for books (1 collision per 7 splits) than for DVDs (1 to 2).

Graphs from G_{14} to G_{19} all have a triangle, with one additional node attached. Again, except for DVDs, splits of recommendations (G_{14} and G_{15}) are more frequent than collisions (G_{18} , G_{19}). For DVDs the most frequent sub-graph of the set is G_{18} (a collision), followed by G_{14} and G_{15} .

A common observation is that simpler graphs, like chains and trees, tend to be more frequent in book recommendation networks, while for DVDs we observe richer and more diverse graphs all with relatively high counts.

Table 6.11 shows larger graph patterns. Various types of collisions are becoming more frequent. For book cascades G_{27} is very frequent, while a version with reversed edges can only be found in DVDs. Graphs G_{34} and G_{35} are the two largest that can be found in recommendation networks from all 4 product groups. Larger DVD cascades tend to be frequent – G_{35} ranks 18 among DVD cascades.

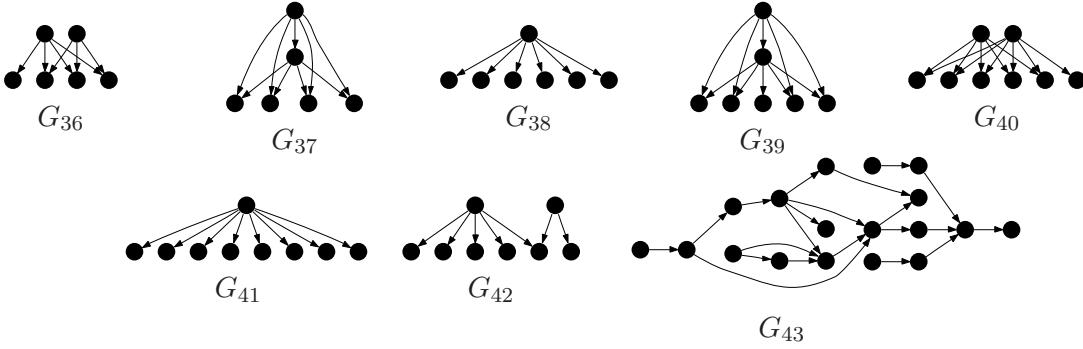


Figure 6.20: Typical classes of cascades. G_{36}, G_{40} : nodes recommending to the same set of people, but not each other. G_{38}, G_{41} : a flat cascade. G_{37}, G_{39} : nodes recommending to same community. G_{43} is an example of a large cascade.

Last, figure 6.20 shows typical classes of cascades. Graphs G_{36} and G_{40} show the case when two people have the same set of friends but do not recommend to each other. A similar case is represented by cascades G_{37} and G_{39} , where the top node recommends to a set of people, and then one of the people in this set purchases and recommends to the same set of people. Flat cascades are also found (G_{38}, G_{41}, G_{42}) – a person recommends, a number of people respond (and purchase a product), but the cascade does not propagate. Graph G_{43} shows cascade that is quite intricate, but which nonetheless occurred 12 times for DVDs.

6.10 Conclusion

Although the retailer may have hoped to boost its revenues through viral marketing, the additional purchases that resulted from recommendations are just a drop in the bucket of sales that occur through the website. Nevertheless, we were able to obtain a number of interesting insights into how viral marketing works that challenge common assumptions made in epidemic and rumor propagation modeling.

Firstly, it is frequently assumed in epidemic models (*e.g.*, SIRS type of models) that individuals have equal probability of being infected every time they interact [Anderson and May, 2002, Bailey, 1975]. Contrary to this we observe that the probability of infection decreases with repeated interaction. Marketers should take heed that providing excessive incentives for customers to recommend products could backfire by weakening the credibility of the very same links they are trying to take advantage of.

Traditional epidemic and innovation diffusion models also often assume that individuals either have a constant probability of “converting” every time they interact with an infected individual [Goldenberg et al., 2001], or that they convert once the fraction of their contacts who are infected exceeds a node specific threshold [Granovetter, 1978]. In both cases, an increasing number of infected contacts results in an increased likelihood of infection. Instead, we find that the probability of purchasing a product increases with the number of recommendations received, but quickly saturates to a constant and relatively low probability. This means individuals are often impervious to the recommendations of their friends, and resist buying items that they do not want.

In network-based epidemic models, extremely highly connected individuals play a very important role. For example, in needle sharing and sexual contact networks these nodes become the “super-spreaders” by

infecting a large number of people. But these models assume that a high degree node has as much of a probability of infecting each of its neighbors as a low degree node does. In contrast, we find that there are limits to how influential high degree nodes are in the recommendation network. As a person sends out more and more recommendations past a certain number for a product, the success per recommendation declines. This would seem to indicate that individuals have influence over a few of their friends, but not everybody they know.

We also presented a simple stochastic model (Section 6.4.3) that allows for the presence of relatively large cascades for a few products, but reflects well the general tendency of recommendation chains to terminate after just a short number of steps. Aggregating such cascades over all the products, we obtain a highly disconnected network, where the largest component grows over time by aggregating typically very small but occasionally fairly large components. We observed that the most popular categories of items recommended within communities in the largest component reflect differing interests between these communities. We presented a model which shows that these smaller and more tightly knit groups tend to be more conducive to viral marketing.

We saw that the characteristics of product reviews and effectiveness of recommendations vary by category and price, with more successful recommendations being made on technical or religious books, which presumably are placed in the social context of a school, workplace or place of worship. A small fraction of the products accounts for a large proportion of the recommendations. Although not quite as extreme in proportions, the number of successful recommendations also varies widely by product. Still, a sizeable portion of successful recommendations were for a product with only one such sale - hinting at a long tail phenomenon.

The premise behind the study of social networks is that interaction leads to complex collective behavior. Cascades are a form of collective behavior that has been studied theoretically, but for which the study of complete, large-scale datasets has been limited. We have shown that cascades exist in a large real-world recommendation dataset, and investigated some of their structural features.

We developed a practical algorithm and set of techniques to illustrate the existence of cascades, and to measure their frequency. On a large real-life dataset we found novel patterns and our experiments showed that most cascades are small, but large bursts can occur. The cascade sizes and depths follow a power law. Cascade behavior varies a lot among different product types. Topologically, most products (books, music, videos) tend to exhibit small and shallow tree-like cascades, while some (DVDs) can exhibit larger, more complex, and farther-reaching patterns of influence with collisions and expansion across communities.

Since viral marketing was found to be in general not as epidemic as one might have hoped, marketers hoping to develop normative strategies for word-of-mouth advertising should analyze the topology and interests of the social network of their customers. Our study has provided a number of new insights which we hope will have general applicability to marketing strategies and to future models of viral information spread.

Chapter 7

Information propagation on the blogosphere

How do blogs cite and influence each other? How do such links evolve? Does the popularity of old blog posts drop exponentially with time? These are some of the questions that we address in this work. Our goal is to build a model that generates realistic cascades, so that it can help us with link prediction and outlier detection.

Blogs (weblogs) have become an important medium of information because of their timely publication, ease of use, and wide availability. In fact, they often make headlines, by discussing and discovering evidence about political events and facts. Often blogs link to one another, creating a publicly available record of how information and influence spreads through an underlying social network. Aggregating links from several blog posts creates a directed graph which we analyze to discover the patterns of information propagation in blogspace, and thereby understand the underlying social network. Not only are blogs interesting on their own merit, but our analysis also sheds light on how rumors, viruses, and ideas propagate over social and computer networks.

Here we report some surprising findings of the blog linking and information propagation structure, after we analyzed one of the largest available datasets, with 45,000 blogs and ≈ 2.2 million blog-postings. We also present a simple model that mimics the spread of information on the blogosphere, and produces information cascades very similar to those found in real life.

7.1 Introduction

Blogs have become an important medium of communication and information on the World Wide Web. Due to their accessible and timely nature, they are also an intuitive source for data involving the spread of information and ideas. By examining linking propagation patterns from one blog post to another, we can infer answers to some important questions about the way information spreads through a social network over the Web. For instance, does traffic in the network exhibit bursty, and/or periodic behavior? After a topic becomes popular, how does interest die off – linearly, or exponentially?

In addition to temporal aspects, we would also like to discover topological patterns in information propagation graphs (cascades). We explore questions like: do graphs of information cascades have common

shapes? What are their properties? What are characteristic in-link patterns for different nodes in a cascade? What can we say about the size distribution of cascades?

Finally, how can we build models that generate realistic cascades?

7.1.1 Summary of findings and contributions

Temporal patterns: For the two months of observation, we found that blog posts do *not* have a bursty behavior; they only have a weekly periodicity. Most surprisingly, the popularity of posts drops with a *power law*, instead of exponentially, that one may have expected. Surprisingly, the exponent of the power law is ≈ -1.5 , agreeing very well with Barabasi's theory of heavy tails in human behavior [Barabási, 2005].

Patterns in the shapes and sizes of cascades and blogs: Almost every metric we measured, followed a power law. The most striking result is that the size distribution of cascades (= number of involved posts), follows a perfect Zipfian distribution, that is, a power law with slope =-2. The other striking discovery was on the shape of cascades. The most popular shapes were the “stars”, that is, a single post with several in-links, but none of the citing posts are themselves cited.

Generating Model: Finally, we design a flu-like epidemiological model. Despite its simplicity, it generates cascades that match several of the above power law properties of real cascades. This model could be useful for link prediction, link-spam detection, and “what-if” scenarios.

7.1.2 Chapter organization

In section 7.2 we briefly survey related work. We introduce basic concepts and terminology in section 7.3. Next, we describe the blog dataset, and discuss the data cleaning steps. We describe temporal link patterns in section 7.5, and continue with exploring the characteristics of the information cascades. We develop and evaluate the Cascade generation model in section 7.6. We discuss implications of our findings in section 7.7, and conclude in section 7.8.

7.2 Connection to temporal modeling and epidemiology

To our knowledge this work presents the first analysis of temporal aspects of blog link patterns, and gives detailed analysis about cascades and information propagation on the blogosphere. As we explore the methods for modeling such patterns, we will refer to concepts involving power laws and burstiness, social networks in the blog domain, and information cascades.

7.2.1 Burstiness and power laws

How often do people create blog posts and links? Extensive work has been published on patterns relating to human behavior, which often generates bursty traffic. Disk accesses, network traffic, web-server traffic all exhibit burstiness. Wang et al. in [Wang et al., 2002] provide fast algorithms for modeling such burstiness. Burstiness is often related to self-similarity, which was studied in the context of World Wide

Web traffic [Crovella and Bestavros, 1997]. Vazquez et al. [Vazquez et al., 2006] demonstrate the bursty behavior in web page visits and corresponding response times.

Self-similarity is often a result of heavy-tailed dynamics. Human interactions may be modeled with networks, and attributes of these networks often follow *power law* distributions [Faloutsos et al., 1999]. Such distributions have a PDF (probability density function) of the form $p(x) \propto x^\gamma$, where $p(x)$ is the probability to encounter value x and γ is the exponent of the power law. In log-log scales, such a PDF gives a straight line with slope γ . For $\gamma < -1$, we can show that the Complementary Cumulative Distribution Function (CCDF) is also a power law with slope $\gamma + 1$, and so is the rank-frequency plot pioneered by Zipf [Zipf, 1949], with slope $1/(1 + \gamma)$. For $\gamma = -2$ we have the standard Zipf distribution, and for other values of γ we have the generalized Zipf distribution.

Human activity also follows periodicities, like daily, weekly and yearly periodicities, often in combination with the burstiness.

7.2.2 Blogs

Most work on modeling link behavior in large-scale on-line data has been done in the domain of blogs and social media [Kumar et al., 2003, Adamic and Glance, 2005, Adar and Adamic, 2005]. The authors note that, while information propagates between blogs, examples of genuine cascading behavior appeared relatively rare. This may, however, be due in part to the Web-crawling and text analysis techniques used to infer relationships among posts [Adar and Adamic, 2005, Gruhl et al., 2004]. Our work here differs in a way that we concentrate solely on the propagation of links, and do not infer additional links from text of the post, which gives us more accurate information.

There are several potential models to capture the structure of the blogosphere. Work on information diffusion based on topics [Gruhl et al., 2004] showed that for some topics, their popularity remains constant in time (“chatter”) while for other topics the popularity is more volatile (“spikes”). Authors in [Kumar et al., 2003] analyze community-level behavior as inferred from blog-rolls – permanent links between “friend” blogs. Analysis based on thresholding as well as alternative probabilistic models of node activation is considered in the context of finding the most influential nodes in a network [Kempe et al., 2003], and for viral marketing [Richardson and Domingos, 2002b]. Such analytical work posits a known network, and uses the model to find the most influential nodes; in the current work we observe real cascades, characterize them, and build generative models for them.

7.2.3 Information cascades and epidemiology

Information cascades are phenomena in which an action or idea becomes widely adopted due to the influence of others, typically, neighbors in some network [Bikhchandani et al., 1992, Goldenberg et al., 2001, Granovetter, 1978]. Cascades on random graphs using a threshold model have been theoretically analyzed [Watts, 2002]. Empirical analysis of the topological patterns of cascades in the context of a large product recommendation network is in [Leskovec et al., 2006b] and [Leskovec et al., 2006a].

The study of epidemics offers powerful models for analyzing the spread of viruses. Our topic propagation model is based on the *SIS* (Susceptible-Infected-Susceptible) model of epidemics [Bailey, 1975]. This models flu-like viruses, where an entity begin as “susceptible”, may become “infected” and infectious, and then heals to become susceptible again. A key parameter is the infection probability β ,

SYMBOL	DESCRIPTION
N	Number of nodes in a cascade
E	Number of edges in a cascade
β	Probability of cascade propagation
t_u	Time that post u was published
Δ	Propagation delay on edge (u, v) , $\Delta = t_u - t_v$

Table 7.1: Table of symbols.

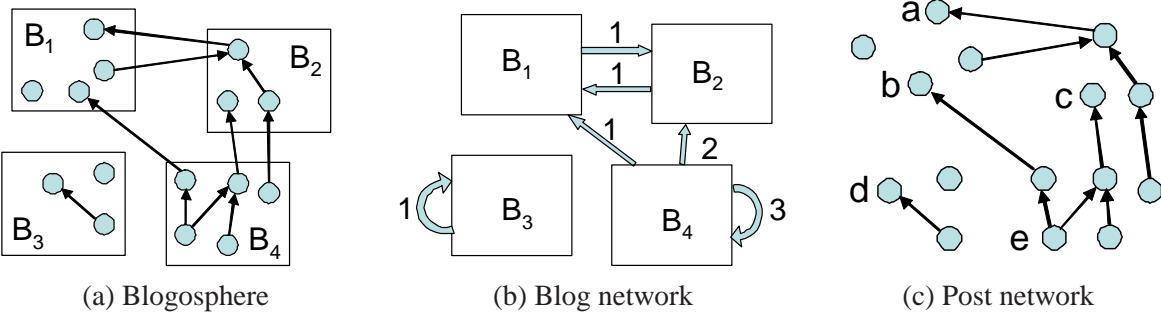


Figure 7.1: The model of the blogosphere (a). Squares represent blogs and circles blog-posts. Each post belongs to a blog, and can contain hyper-links to other posts and resources on the web. We create two networks: a weighted blog network (b) and a post network (c). Nodes a, b, c, d are *cascade initiators*, and node e is a *connector*.

that is, the probability of a disease transmission in a single contact. Of high interest is the *epidemic threshold*, that is, the critical value of β , above which the virus will spread and create an epidemic, as opposed to becoming extinct. There is a huge literature on the study of epidemics on full cliques, homogeneous graphs, infinite graphs (see [Hethcote, 2000] for a survey), with recent studies on power law networks [Eguiluz and Klemm, 2002] and arbitrary networks [Wang et al., 2003].

7.3 Preliminaries

In this section we introduce terminology and basic concepts regarding the blogosphere and information cascades.

Blogs (weblogs) are web sites that are updated on a regular basis. Blogs have the advantage of being easy to access and update, and have come to serve a variety of purposes. Often times individuals use them for online diaries and social networking, other times news sites have blogs for timely stories. Blogs are composed of posts that typically have room for comments by readers – this gives rise to discussion and opinion forums that are not possible in the mass media. Also, blogs and posts typically link each other, as well as other resources on the Web. Thus, blogs have become an important means of transmitting information. The influence of blogs was particularly relevant in the 2004 U.S. election, as they became sources for campaign fundraising as well as an important supplement to the mainstream media [Adamic and Glance, 2005]. The blogosphere has continued to expand its influence, so understanding the ways in which information is transmitted among blogs is important to developing concepts of present-day communication.

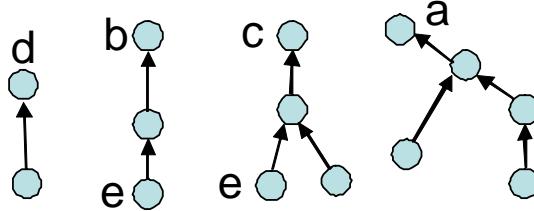


Figure 7.2: Cascades extracted from Figure 7.1. Cascades represent the flow of information through nodes in the network. To extract a cascade we begin with an initiator with no out-links to other posts, then add nodes with edges linking to the initiator, and subsequently nodes that link to any other nodes in the cascade.

We model two graph structures emergent from links in the blogosphere, which we call the *Blog network* and the *Post network*. Figure 7.1 illustrates these structures. Blogosphere is composed of blogs, which are further composed of posts. Posts then contain links to other posts and resources on the web.

From Blogosphere (a), we obtain the Blog network (b) by collapsing all links between blog posts into weighted edges between blogs. A directed blog-to-blog edge is weighted with the total number of links occurring between posts in source blog pointing to posts in destination blog. From the Blog network we can infer a social network structure, under the assumption that blogs that are “friends” link each other often.

In contrast, to obtain the Post network (c), we ignore the posts’ parent blogs and focus on the link structure. Associated with each post is also the time of the post, so we label the edges in Post network with the time difference Δ between the source and the destination posts. Let t_u and t_v denote post times of posts u and v , where u links to v , then the link time $\Delta = t_u - t_v$. Note $\Delta > 0$, since a post can not link into the future and there are no self-edges.

From the Post network, we extract information cascades, which are induced subgraphs by edges representing the flow of information. A cascade (also known as conversation tree) has a single starting post called the *cascade initiator* with no out-links to other posts (e.g., nodes a, b, c, d in Figure 7.1(c)). Posts then join the cascade by linking to the initiator, and subsequently new posts join by linking to members within the cascade, where the links obey time order ($\Delta > 0$). Figure 7.2 gives a list of cascades extracted from Post network in Figure 7.1(c). Since a link points from the follow-up post to the existing (older) post, influence propagates following the reverse direction of the edges.

We also define a *non-trivial* cascade to be a cascade containing at least two posts, and therefore a *trivial cascade* is an isolated post. Figure 7.2 shows all non-trivial cascades in Figure 7.1(c), but not the two trivial cascades. Cascades form two main shapes, which we will refer to as *stars* and *chains*. A star occurs when a single center post is linked by several other posts, but the links do not propagate further. This produces a wide, shallow tree. Conversely, a chain occurs when a root is linked by a single post, which in turn is linked by another post. This creates a deep tree that has little breadth. As we will later see most cascades are somewhere between these two extreme points. Occasionally separate cascades might be joined by a single post – for instance, a post may summarize a set of topics, or focus on a certain topic and provide links to different sources that are members of independent cascades. The post merging the cascades is called a *connector node*. Node e in Figure 7.2(c) is a connector node. It appears in two cascades by connecting cascades starting at nodes b and c .

7.4 Experimental setup

7.4.1 Dataset description

We extracted our dataset from a larger set which contains 21.3 million posts from 2.5 million blogs from August and September 2005 [Glance et al., 2005]. Our goal here is to study temporal and topological characteristics of information propagation on the blogosphere. This means we are interested in blogs and posts that actively participate in discussions, so we biased our dataset towards the more active part of the blogosphere.

We collected our dataset using the following procedure. We started with a list of the most-cited blog posts in August 2005. For all posts we traversed the full conversation tree forward and backward following post's in- and out-links. For practical reasons we limited the depth of such conversation trees to 100 and the maximum number of links followed from a single post to 500. This process gave us a set of posts participating in conversations. From the posts we extracted a list of all blogs. This gave us a set of about 45,000 active blogs. Now, we went back to the original dataset and extracted all posts coming from this set of active blogs.

This process produced a dataset of 2,422,704 posts from 44,362 blogs gathered over a two-month period from beginning of August to end of September 2005. There are the total of 4,970,687 links in the dataset out of which 245,404 are among the posts of our dataset and the rest point to other resources (*e.g.*, images, press, news, web-pages). For each post in the dataset we have the following information: unique Post ID, the URL of the parent blog, Permalink of the post, Date of the post, post content (html), and a list of all links that occur in the post's content. Notice these posts are not a random sample of all posts over the two month period but rather a set of posts biased towards active blogs participating in conversations (by linking to other posts/blogs).

In Figure 7.3 we plot the number of posts per day over the span of our dataset. The periodicities in traffic on a weekly basis will be discussed in section 7.5. Notice that our dataset has no “missing past” problem, *i.e.*, the starting points of conversation are not missing due to the beginning of data collection, since we followed the conversation all the way to its starting point and thus obtained complete conversations. The posts span the period from July to September 2005 (90 days), while the majority of the data comes from August and September. The July posts in the dataset are parts of conversations that were still active in August and September.

7.4.2 Data preparation and cleaning

We represent the data as a cluster graph where clusters correspond to blogs, nodes in the cluster are posts from the blog, and hyper-links between posts in the dataset are represented as directed edges. Before analysis, we cleaned the data to most clearly represent the structures of interest.

Only consider out-links to posts in the dataset. We removed links that point to posts outside our dataset or other resources on the web (images, movies, other web-pages). The major reason for this is that we only have time-stamps for the posts in the dataset while we know nothing about creation time of URLs outside the dataset, and thus we cannot consider these links in our temporal analysis.

Use time resolution of one day. While posts in blogspace are often labeled with complete time-stamps, many posts in our dataset do not have a specific time stamp but only the date is known. Additionally,

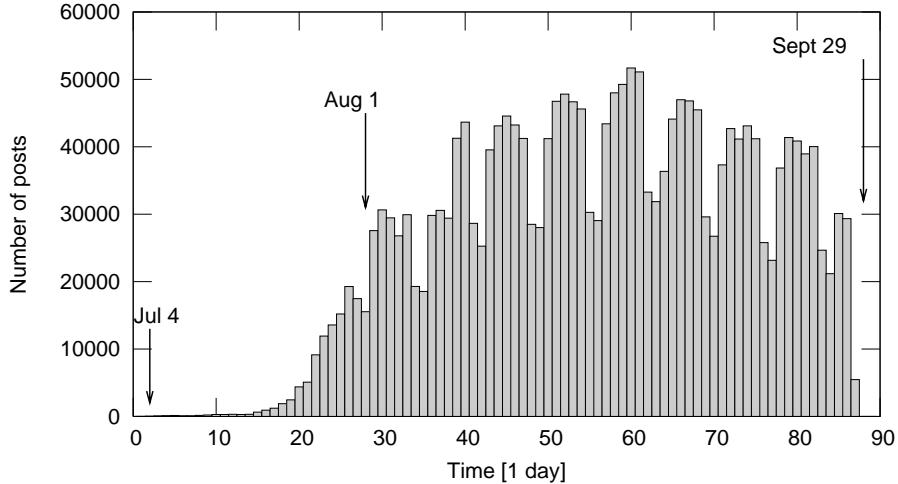


Figure 7.3: Number of posts by day over the three-month period.

there are challenges in using time stamps to analyze emergent behaviors on an hourly basis, because posts are written in different time zones, and we do not normalize for this. Using a coarser resolution of one day serves to reduce the time zone effects. Thus, in our analysis the time differences are aggregated into 24-hour bins.

Remove edges pointing into the future. Since a post cannot link to another post that has not yet been written, we remove all edges pointing into the future. The cause may be human error, post update, an intentional back-post, or time zone effects; in any case, such links do not represent information diffusion.

Remove self edges. Again, self edges do not represent information diffusion. However, we do allow a post to link to another post in the same blog.

7.5 Observations, patterns and laws

Next we present our experiments and observations on the blog and post network topology and cascading patterns of information diffusion on the blogosphere.

7.5.1 Temporal dynamics of posts and links

Traffic in blogosphere is not uniform; therefore, we consider traffic patterns when analyzing influence in the temporal sense. As Figure 7.3 illustrates, there is a seven-day periodicity. Further exploring the weekly patterns, Figure 7.4 shows the number of posts and the number of blog-to-blog links for different days of the week, aggregated over the entire dataset. Posting and blog-to-blog linking patterns tend to have a *weekend effect* of sharply dropping off at weekends.

Next, we examine how a post's popularity grows and declines over time. We collect all in-links to a post and plot the number of links occurring after each day following the post. This creates a curve that

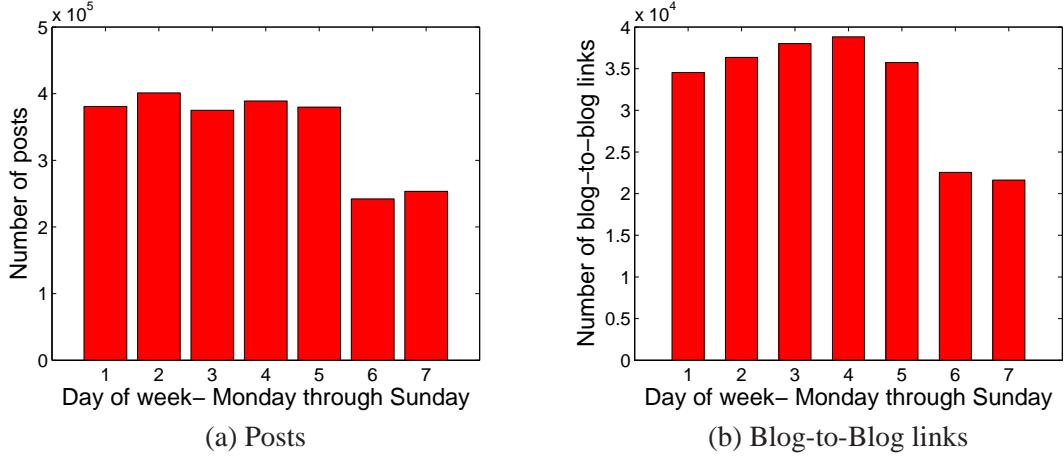


Figure 7.4: Activity counts (number of posts and number of links) per day of week, from Monday to Sunday, summed over entire dataset.

indicates the rise and fall of popularity. By aggregating over a large set of posts we obtain a more general pattern.

Top left plot of Figure 7.5 shows number of in-links for each day following a post for all posts in the dataset, while top right plot shows the in-link patterns for Monday posts only (in order to isolate the weekly periodicity). It is clear that the most links occur on the first 24 hours after the post, after that the popularity generally declines. However, in the top right plot, we note that there are “spikes” occurring every seven days, each following Monday. It almost appears as if there is compensatory behavior for the sparse weekend links. However, this is not the case. Mondays do not have an unusual number of links; Monday only appears to spike on these graphs because the natural drop-off of popularity in the following days allows Monday to tower above its followers.

Thus, fitting a general model to the drop-off graphs may be problematic, since we might obtain vastly different parameters across posts simply because they occur at different times during the week. Therefore, we smooth the in-link plots by applying a weighting parameter to the plots separated by day of week. For each delay Δ on the horizontal axis, we estimate the corresponding day of week d , and we prorate the count for Δ by dividing it by $l(d)$, where $l(d)$ is the percent of blog links occurring on day of week d .

This weighting scheme normalizes the curve such that days of the week with less traffic are bumped up further to meet high traffic days, simulating a popularity drop-off that might occur if posting and linking behavior were uniform throughout the week. A smoothed version of the post drop-offs is shown in the middle row of Figure 7.5.

We fit the power law distribution with a cut-off in the tail (bottom row). We fit on 30 days of data, since most posts in the graph have complete in-link patterns for the 30 days following publication. We performed the fitting over all posts and for all days of the week separately, and found a stable power law exponent of around -1.5 , which is exactly the value predicted by the model where the bursty nature of human behavior is a consequence of a decision based queuing process [Barabási, 2005] – when individuals execute tasks based on some perceived priority, the timing of the tasks is heavy tailed, with most tasks being rapidly executed, whereas a few experience very long waiting times.

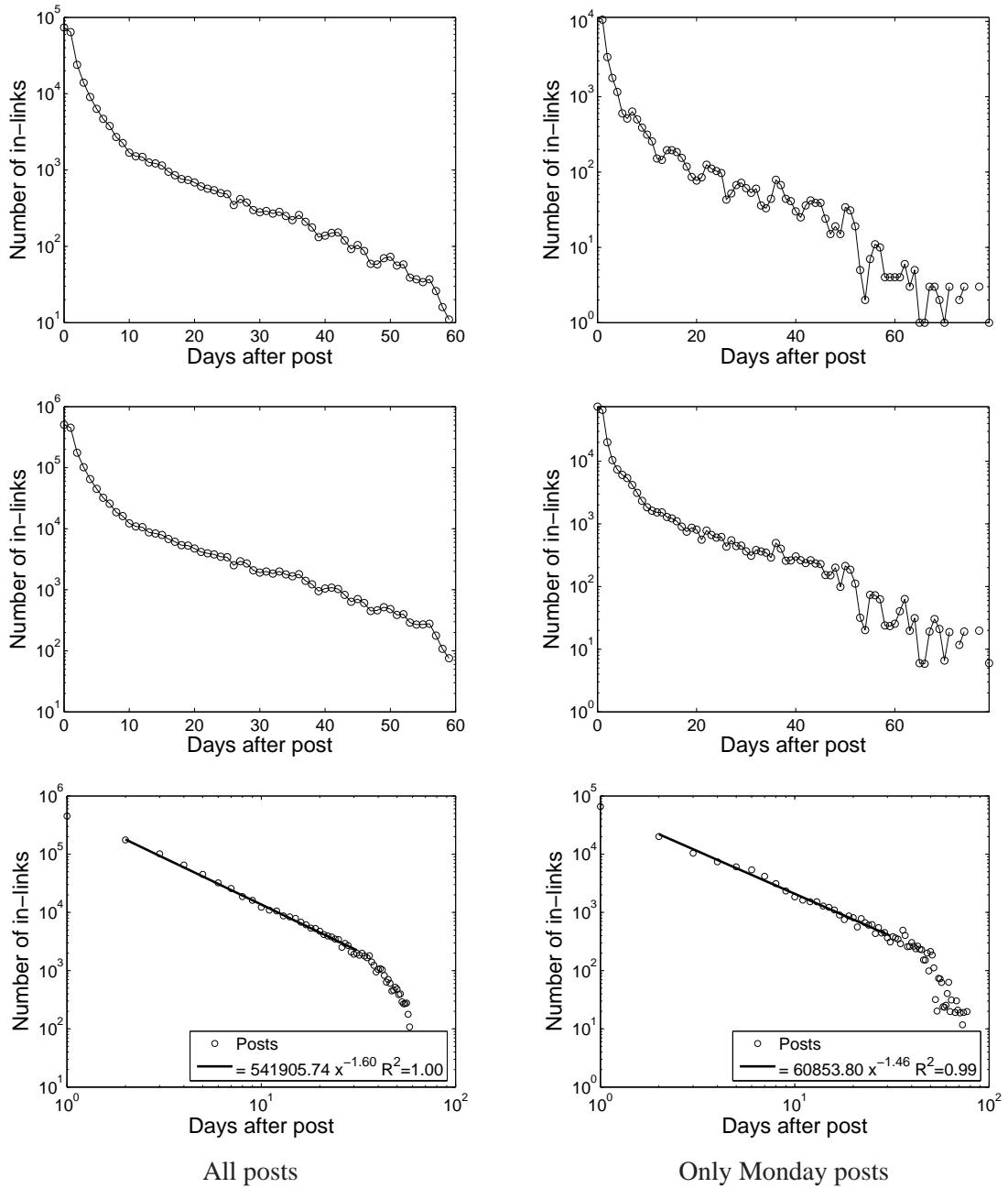


Figure 7.5: Number of in-links vs. the days after the post in log-linear scale; when considering all posts (top left), only Monday posts (top right). After removing the day-of-the week effects (middle row). Power law fit to the data with exponents -1.6 and -1.46 (bottom row).

Observation 7.5.1. *The probability that a post u written at time $t(u)$ acquires a link at time $t(u) + \Delta$ is:*

$$p(t(u) + \Delta) \propto \Delta^{-1.5}$$

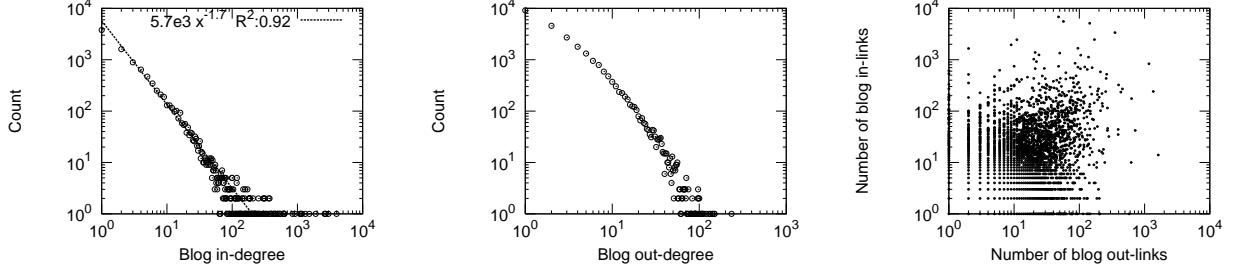


Figure 7.6: (left, middle) In- and out-degree distributions of the Blog network; (right) the scatter plot of the number of in- and out-links of the blogs.

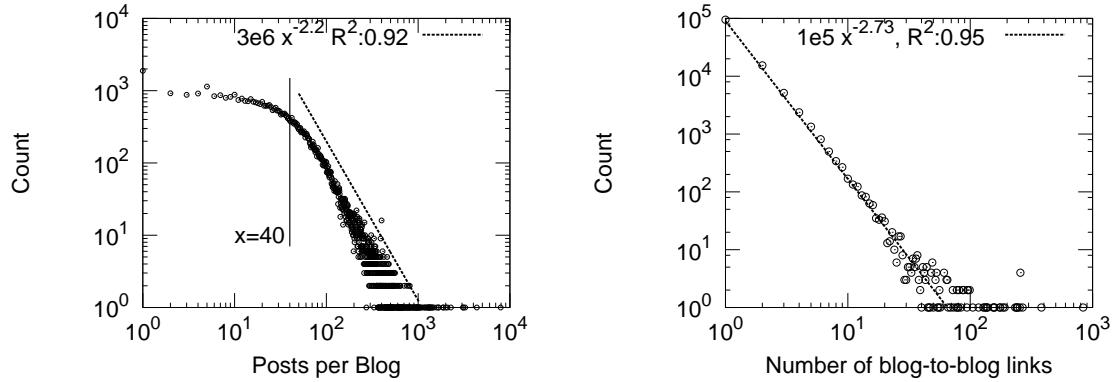


Figure 7.7: Distribution of the number of posts per blog (a); Distribution of the number of blog-to-blog links, *i.e.*, the distribution over the Blog network edge weights (b).

7.5.2 Blog network topology

The first graph we consider is the Blog network. As illustrated in Figure 7.1(c), every node represents a blog and there is a weighted directed edge between blogs u and v , where the weight of the edge corresponds to the number of posts from blog u linking to posts at blog v . The network contains 44,356 nodes and 122,153 edges. The sum of all edge weights is the number of all post to post links (245,404). Connectivity-wise, half of the blogs belong to the largest connected component and the other half are isolated blogs.

We show the in- and out-degree distribution in Figure 7.6. Notice they both follow a heavy-tailed distribution. The in-degree distribution has a very shallow power law exponent of -1.7 , which suggests strong rich-get-richer phenomena. One would expect that popular active blogs that receive lots of in-links also sprout many out-links. Intuitively, the attention (number of in-links) a blog gets should be correlated with its activity (number of out-links). This does not seem to be the case. The correlation coefficient between a blog's number of in- and out-links is only 0.16 , and the scatter plot in Figure 7.6 suggests the same.

The number of posts per blog, as shown in Figure 7.7(a), follows a heavy-tailed distribution. The deficit of blogs with low number of posts and the knee at around 40 posts per blog can be explained by the fact that we are using a dataset biased towards active blogs. However, our biased sample of the blogs still maintains the power law in the number of blog-to-blog links (edge weights of the Blog network) as shown in 7.7(b). The power law exponent is -2.7 .

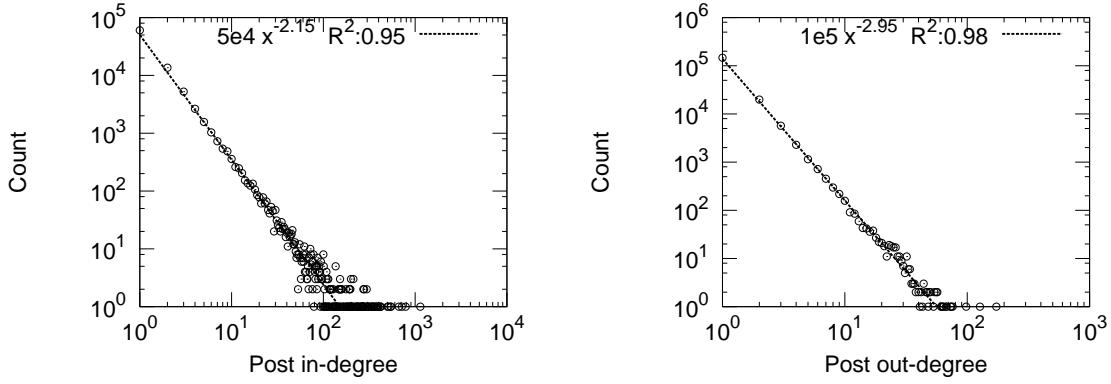


Figure 7.8: Post network in- and out-degree distribution.

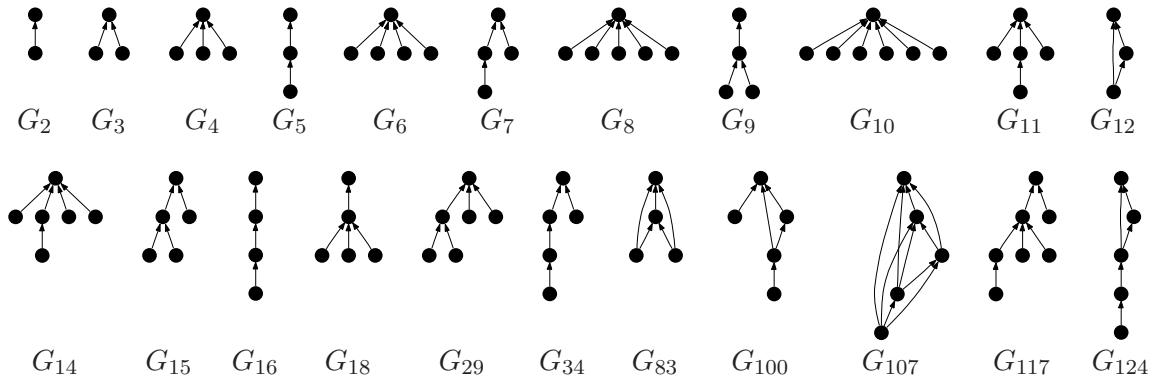


Figure 7.9: Common cascade shapes ordered by the frequency. Cascade with label G_r has the frequency rank r .

7.5.3 Post network topology

In contrast to Blog network the Post network is very sparsely connected. It contains 2.2 million nodes and only 205,000 edges. 98% of the posts are isolated, and the largest connected component accounts for 106,000 nodes, while the second largest has only 153 nodes. Figure 7.8 shows the in- and out-degree distributions of the Post network which follow a power law with exponents -2.1 and -2.9 , respectively.

7.5.4 Patterns in the cascades

We continue with the analysis of the topological aspects of the information cascades formed when certain posts become popular and are linked by the other posts. We are especially interested in how this process propagates, how large are the cascades it forms, and as it will be shown later, what are the models that mimic cascading behavior and produce realistic cascades.

Cascades are subgraphs of the Post network that have a single root post, are time increasing (source links an existing post), and present the propagation of the information from the root to the rest of the cascade.

Given the Post network we extracted all information cascades using the following procedure. We found all cascade initiator nodes, *i.e.*, nodes that have zero out-degree, and started following their in-links. This process gives us a directed acyclic graph with a single root node. As illustrated in Figure 7.2 it can happen that two cascades merge, *e.g.*, a post gives a summary of multiple conversations (cascades). For cascades that overlap our cascade extraction procedure will extract the nodes below the connector node multiple times (since they belong to multiple cascades). To obtain the examples of the common shapes and count their frequency we used the algorithms as described in [Leskovec et al., 2006b].

Common cascade shapes

First, we give examples of common Post network cascade shapes in Figure 7.9. A node represents a post and the influence flows from the top to the bottom. The top post was written first, other posts linking to it, and the process propagates. Graphs are ordered by frequency and the subscript of the label gives frequency rank. Thus, G_{124} is 124th most frequent cascade with 11 occurrences.

We find the total of 2,092,418 cascades, and 97% of them are trivial cascades (isolated posts), 1.8% are smallest non-trivial cascades (G_2), and the remaining 1.2% of the cascades are topologically more complex.

Most cascades can essentially be constructed from instances of stars and trees, which can model more complicated behavior like that shown in Figure 7.9. Cascades tend to be wide, and not too deep. Structure G_{107} , which we call a *cite-all chain*, is especially interesting. Each post in a chain refers to every post before it in the chain.

We also find that the cascades found in the graph tend to take certain shapes preferentially. Also notice that cascade frequency rank does not simply decrease as a function of the cascade size. For example, as shown on Figure 7.9, a 4-star (G_4) is more common than a chain of 3 nodes (G_5). In general stars and shallow bursty cascades are the most common type of cascades.

Cascade topological properties

What is the common topological pattern in the cascades? We next examine the general cascade behavior by measuring and characterizing the properties of real cascades.

First we observe the degree distributions of the cascades. This means that from the Post network we extract all the cascades and measure the overall degree distribution. Essentially we work with a *bag of cascades*, where we treat a cascade as separate disconnected sub-graph in a large network.

Figure 7.10(a) plots the out-degree distribution of the bag of cascades. Notice the cascade out-degree distribution is truncated, which is the result of not perfect link extraction algorithm and the upper bound on the post out-degree (500).

Figure 7.10(b) shows the in-degree distribution of the bag of cascades, and (c) plots the in-degree distribution of nodes at level L of the cascade. A node is at level L if it is L hops away from the root (cascade initiator) node. Notice that the in-degree exponent is stable and does not change much given the level in the cascade. This means that posts still attract attention (get linked) even if they are somewhat late in the cascade and appear towards the bottom of it.

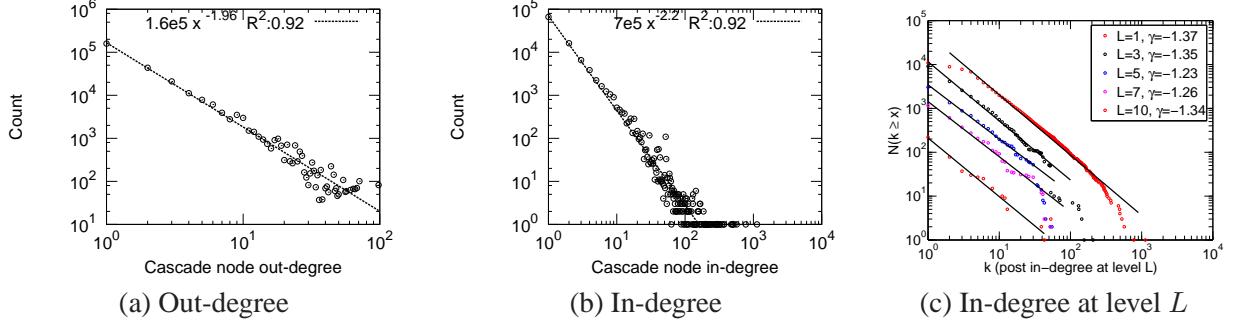


Figure 7.10: Out- and in-degree distribution over all cascades extracted from the Post network (a,b) and the in-degree distribution at level L of the cascade. Note all distributions are heavy tailed and the in-degree distribution is remarkably stable over the levels.

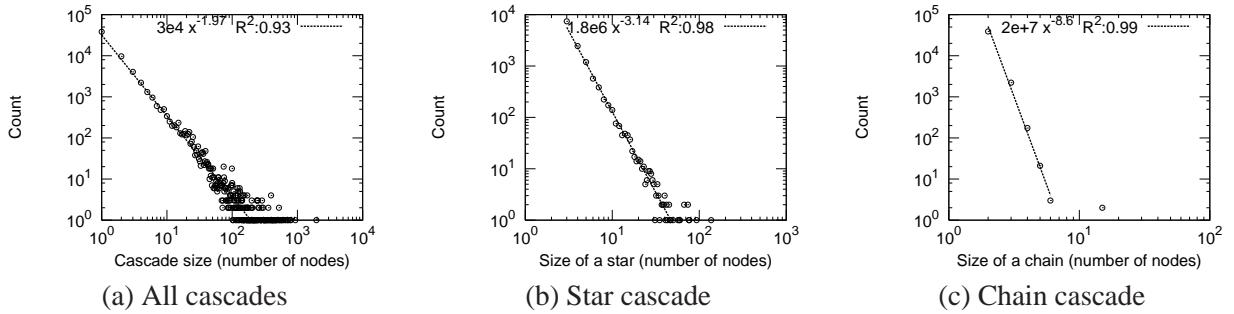


Figure 7.11: Size distribution over all cascades (a), only stars (b), and chains (c). They all follow heavy tailed distributions with increasingly steeper slopes.

Next, we ask what distribution do cascade sizes follow? Does the probability of observing a cascade on N nodes decreases exponentially with N ? We examine the *Cascade Size Distributions* over the bag of cascades extracted from the Post network. We consider three different distributions: over all cascade size distribution, and separate size distributions of star and chain cascades. We chose stars and chains since they are well defined, and given the number of nodes in the cascade, there is no ambiguity in the topology of a star or a chain.

Figure 7.11 gives the Cascade Size Distribution plots. Notice all follow a heavy-tailed distribution. We fit a power law distribution and observe that overall cascade size distribution has power law exponent of ≈ -2 (Figure 7.11(a)), stars have ≈ -3.1 (Figure 7.11(b)), and chains are small and rare and decay with exponent ≈ -8.5 (Fig. 7.11(c)). Also notice there are outlier chains (Fig. 7.11(c)) that are longer than expected. We attribute this to possible flame wars between the blogs, where authors publish posts and always refer to the last post of the other author. This creates chains longer than expected.

Observation 7.5.2. *Probability of observing a cascade on N nodes follows a Zipf distribution:*

$$p(N) \propto N^{-2}$$

As suggested by Figure 7.9 most cascades follow tree-like shapes. To further verify this we examine how the diameter, defined as the length of the longest undirected path in the cascade, and the relation between the number of nodes and the number of edges in the cascade change with the cascade size in Figure 7.12.

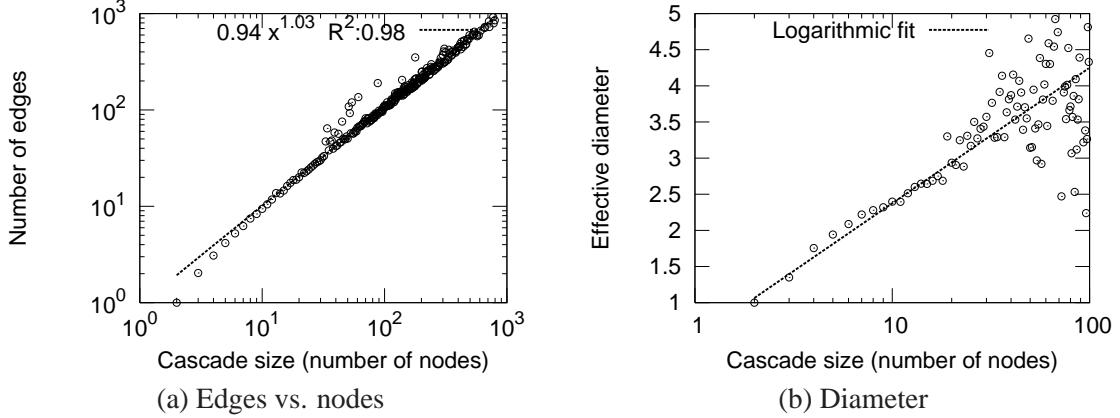


Figure 7.12: Diameter and the number of edges vs. the cascade size. Notice that diameter increases logarithmically with the cascade size, while the number of edges basically grows linearly with the cascade size. This suggests cascades are mostly tree-like structures.

This gives further evidence that the cascades are mostly tree-like. We plot the number of nodes in the cascade vs. the number of edges in the cascade in Figure 7.12(a). Notice the number of edges E in the cascade increases almost linearly with the number of nodes N ($E \propto N^{1.03}$). This suggests that the average degree in the cascade remains constant as the cascade grows, which is a property of trees and stars. Next, we also measure cascade diameter vs. cascade size (Figure 7.12(b)). We plot on linear-log scales and fit a logarithmic function. Notice the diameter increases logarithmically with the size of the cascade, which means the cascade needs to grow exponentially to gain linear increase in diameter, which is again a property of the balanced trees and very sparse graphs.

Collisions of cascades

By the definition we adopt in this chapter, the cascade has a single initiator node, but in real life one would also expect that cascades collide and merge. There are connector nodes which are the first to bring together separate cascades. As the cascades merge, all the nodes below the connector node now belong to multiple cascades. We measure the distribution over the connector nodes and the nodes that belong to multiple cascades.

First, we consider only the connector nodes and plot the distribution over how many cascades a connector joins (Figure 7.13(a)). We only consider nodes with out-degree greater than 1, since nodes with out-degree 1 are trivial connectors – they are connecting the cascade they belong to. But there are still posts that have out-degree greater than 1, and connect only one cascade. These are the posts that point multiple out-links inside the same cascade (e.g. G_{12} and G_{107} of Figure 7.9). The dip at the number of joined cascades equal to 1 in Figure 7.13(a) gives the number of such nodes.

As cascades merge, all the nodes that follow belong to multiple cascades. Figure 7.13(b) gives the distribution over the number of cascades a node belongs to. Here we consider all the nodes and find out that 98% of all nodes belong to a single cascade, and the rest of distribution follows a power law with exponent -2.2 .

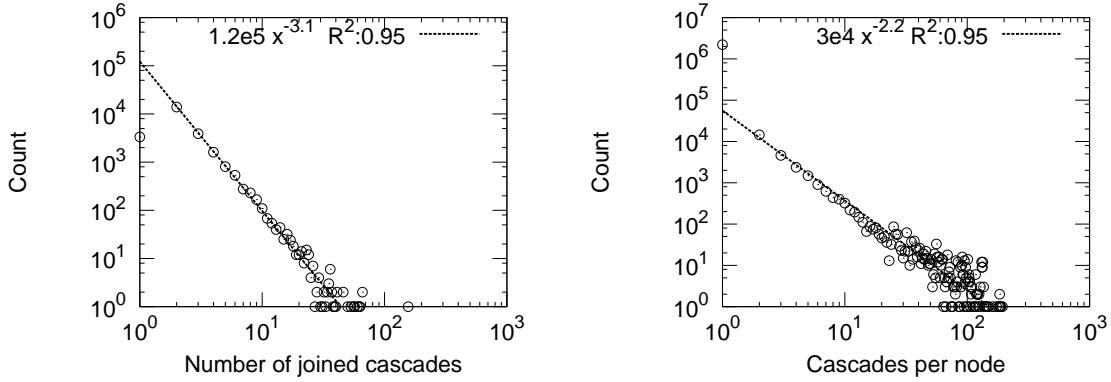


Figure 7.13: Distribution of joined cascades by the connector nodes (a). We only consider nodes with out-degree greater than 1. Distribution of a number of cascades a post belongs to (b); 98% of posts belong to a single cascade.

7.6 Proposed model and insights

What is the underlying hidden process that generates cascades? Our goal here is to find a generative model that generates cascades with properties observed in section 7.5.4 (Figures 7.10 and 7.11). We aim for a simple and intuitive model with the least possible number of parameters.

7.6.1 Cascade generation model

We present a conceptual model for generating information cascades that produces cascade graphs matching several properties of real cascades. Our model is intuitive and requires only a single parameter that corresponds to how interesting (easy spreading) are the conversations in general on the blogosphere.

Intuitively, cascades are generated by the following principle. A post is posted at some blog, other bloggers read the post, some create new posts, and link the source post. This process continues and creates a cascade. One can think of cascades being a graph created by the spread of the virus over the Blog network. This means that the initial post corresponds to infecting a blog. As the cascade unveils, the virus (information) spreads over the network and leaves a trail. To model this process we use a single parameter β that measures how infectious are the posts on the blogosphere. Our model is very similar to the SIS (susceptible – infected – susceptible) model from the epidemiology [Hethcote, 2000].

Next, we describe the model. Each blog is in one of two states: *infected* or *susceptible*. If a blog is in the infected state this means that the blogger just posted a post, and the blog now has a chance to spread its influence. Only blogs in the susceptible (not infected) state can get infected. When a blog successfully infects another blog, a new node is added to the cascade, and an edge is created between the node and the source of infection. The source immediately recovers, *i.e.*, a node remains in the infected state only for one time step. This gives the model the ability to infect a blog multiple times, which corresponds to multiple posts from the blog participating in the same cascade.

More precisely, a single cascade of the *Cascade generation model* is generated by the following process.

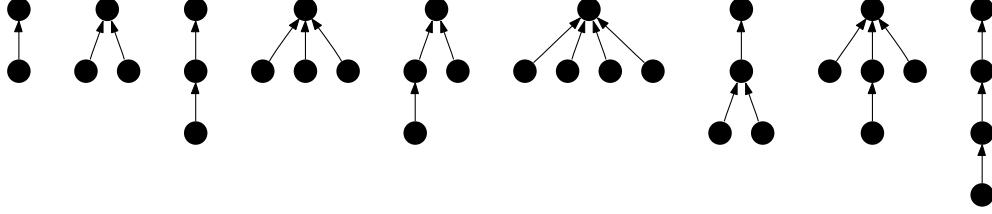


Figure 7.14: Top 10 most frequent cascades as generated by the Cascade generation model. Notice similar shapes and frequency ranks as in Figure 7.9.

- (i) Uniformly at random pick blog u in the Blog network as a starting point of the cascade, set its state to *infected*, and add a new node u to the cascade graph.
- (ii) Blog u that is now in infected state, infects each of its uninfected directed neighbors in the Blog network independently with probability β . Let $\{v_1, \dots, v_n\}$ denote the set of infected neighbors.
- (iii) Add new nodes $\{v_1, \dots, v_n\}$ to the cascade and link them to node u in the cascade.
- (iv) Set state of node u to not infected. Continue recursively with step (ii) until no nodes are infected.

We make a few observations about the proposed model. First, note that the blog immediately recovers and thus can get infected multiple times. Every time a blog gets infected a new node is added to the cascade. This accounts for multiple posts from the blog participating in the same cascade. Second, we note that in this version of the model we do not try to account for topics or model the influence of particular blogs. We assume that all blogs and all conversations have the same value of the parameter β . Third, the process as described above generates cascades that are trees. This is not big limitation since we observed that most of the cascades are trees or tree-like. In the spirit of our notion of cascade we assume that cascades have a single starting point, and do not model for the collisions of the cascades.

7.6.2 Validation of the model

We validate our model by extensive numerical simulations. We compare the obtained cascades towards the real cascades extracted from the Post network. We find that the model matches the cascade size and degree distributions.

We use the real Blog network over which we propagate the cascades. Using the Cascade generation model we also generate the same number of cascades as we found in Post network (≈ 2 million). We tried several values of β parameter, and at the end decided to use $\beta = 0.025$. This means that the probability of cascade spreading from the infected to an uninfected blog is 2.5%. We simulated our model 10 times, each time with a different random seed, and report the average.

First, we show the top 10 most frequent cascades (ordered by frequency rank) as generated by the Cascade generation model in Figure 7.14. Comparing them to most frequent cascades from Figure 7.9 we notice that top 7 cascades are matched exactly (with an exception of ranks of G_4 and G_5 swapped), and remaining cascades can also be found in real data.

Next, we show the results on matching the cascade size and degree distributions in Figure 7.15. We plot the true distributions of the cascades extracted from the Post network with dots, and the results of

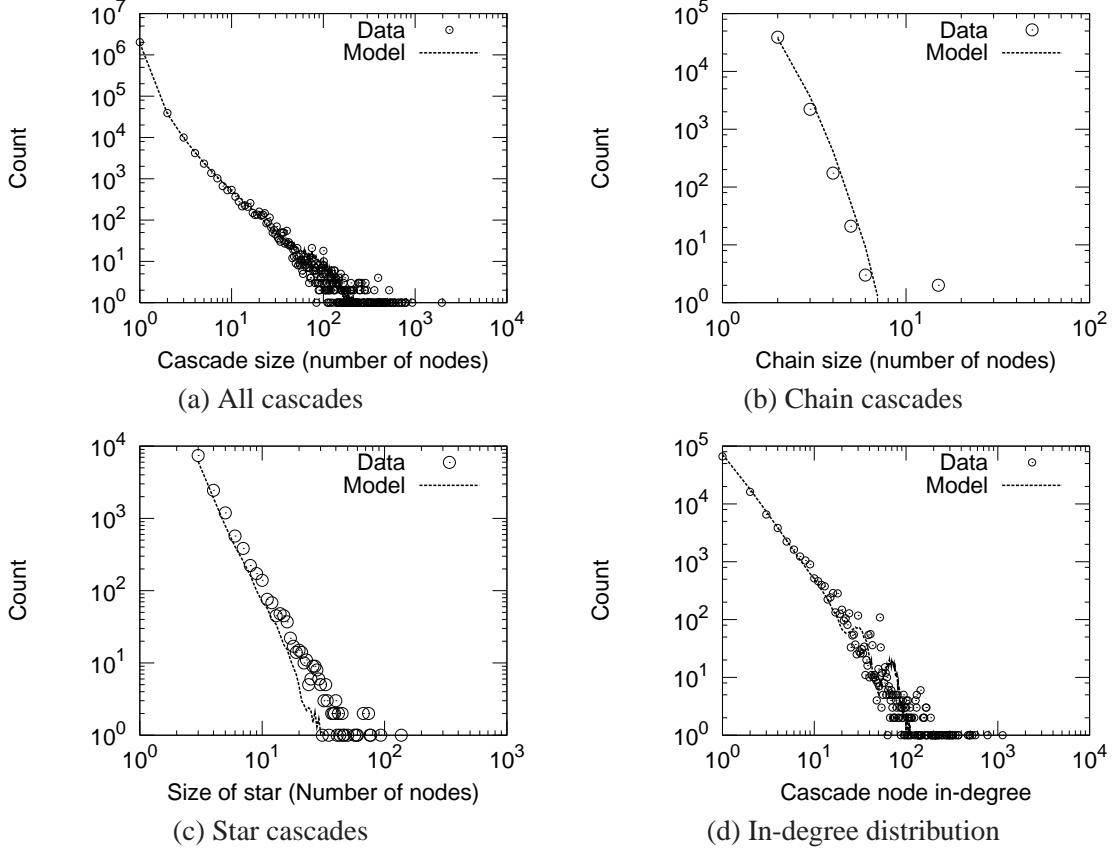


Figure 7.15: Comparison of the true data and the model. We plotted the distribution of the true cascades with circles and the estimate of our model with dashed line. Notice remarkable agreement between the data and the prediction of our simple model.

our model are plotted with a dashed line. We compare four properties of cascades: (a) overall cascade size distribution, (b) size distribution of chain cascades, (c) size distribution of stars, and (d) in-degree distribution over all cascades.

Notice a very good agreement between the reality and simulated cascades in all plots. The distribution over of cascade sizes is matched best. Chains and stars are slightly under-represented, especially in the tail of the distribution where the variance is high. The in-degree distribution is also matched nicely, with an exception of a spike that can be attributed to a set of outlier blogs all with in-degree 52. Cascades generated by the Cascade generation model are all trees, and thus the out-degree for every node is 1.

7.6.3 Variations of the model

We also experimented with other, more sophisticated versions of the model. Namely, we investigated various strategies of selecting a starting point of the cascade, and using edge weights (number of blog-to-blog links) to further boost cascades.

We considered selecting a cascade starting blog based on the blog in-degree, in-weight or the number of posts. We experimented with variants where the probability β of propagating via a link is not constant

but also depends on the weight of the link (number of references between the blogs). We also considered versions of the model where the probability β exponentially decays as the cascade spreads away from the origin.

We found out that choosing a cascade starting blog in a biased way results in too large cascades and non-heavy tailed distributions of cascade sizes. Intuitively, this can be explained by the fact that popular blogs are in the core of the Blog network, and it is very easy to create large cascades when starting in the core. A similar problem arises when scaling β with the edge weight. This can also be explained by the fact that we are not considering specific topics and associate each edge with a topic (some blog-to-blog edges may be very topic-specific) and thus we allow the cascade to spread over all edges regardless of the particular reason (the topic) that the edge between the blogs exists. This is especially true for blogs like BoingBoing (www.boingboing.net) that are very general and just a collection of “wonderful things”.

7.7 Discussion

Our finding that the popularity of posts drops off with a power law distribution is interesting since intuition might lead one to believe that people would “forget” a post topic in an exponential pattern. However, since linking patterns are based on the behaviors of individuals over several instances, much like other real-world patterns that follow power laws such as traffic to Web pages and scientists’ response times to letters [Vazquez et al., 2006], it is reasonable to believe that a high number of individuals link posts quickly, and later linkers fall off with a heavy-tailed pattern.

Our findings have potential applications in many areas. One could argue that the conversation mass metric, defined as the total number of posts in all conversation trees below the point in which the blogger contributed, summed over all conversation trees in which the blogger appears, is a better proxy for measuring influence. This metric captures the mass of the total conversation generated by a blogger, while number of in-links captures only direct responses to the blogger’s posts.

For example, we found that BoingBoing, which a very popular blog about amusing things, is engaged in many cascades. Actually, 85% of all BoingBoing posts were cascade initiators. The cascades generally did not spread very far but were wide (*e.g.*, G_{10} and G_{14} in Fig. 7.9). On the other hand 53% of posts from a political blog MichelleMalkin (www.michellemalkin.com) were cascade initiators. But the cascade here were deeper and generally larger (*e.g.*, G_{117} in Fig. 7.9) than those of BoingBoing.

7.8 Conclusion

We analyzed one of the largest available collections of blog information, trying to find how blogs behave and how information propagates through the blogosphere. We studied two structures, the “Blog network” and the “Post network”. Our contributions are two-fold: (a) The discovery of a wealth of temporal and topological patterns and (b) the development of a generative model that mimics the behavior of real cascades. In more detail, our findings are summarized as follows:

- *Temporal Patterns:* The decline of a post’s popularity follows a power law. The slope is ≈ -1.5 , the slope predicted by a very recent theory of heavy tails in human behavior [Barabási, 2005]

- *Topological Patterns:* Almost any metric we examined follows a power law: size of cascades, size of blogs, in- and out-degrees. To our surprise, the number of in- and out-links of a blog are not correlated. Finally, stars and chains are basic components of cascades, with stars being more common.
- *Generative model:* Our idea is to reverse-engineer the underlying social network of blog-owners, and to treat the influence propagation between blog-posts as a flu-like virus, that is, the SIS model in epidemiology. Despite its simplicity, our model generates cascades that match very well the real cascades with respect to in-degree distribution, cascade size distribution, and popular cascade shapes.

Chapter 8

Outbreak and cascade detection

Given a water distribution network, where should we place sensors to quickly detect contaminants? Or, which blogs should we read to avoid missing important stories?

These seemingly different problems share common structure: Outbreak detection can be modeled as selecting nodes (sensor locations, blogs) in a network, in order to detect the spreading of a virus or information as quickly as possible.

In this chapter we present a general methodology for near optimal sensor placement in these and related problems. We demonstrate that many realistic outbreak detection objectives (*e.g.*, detection likelihood, population affected) exhibit the property of “submodularity”. We exploit submodularity to develop an efficient algorithm that scales to large problems, achieving near optimal placements, while being 700 times faster than a simple greedy algorithm. We also derive online bounds on the quality of the placements obtained by *any* algorithm. Our algorithms and bounds also handle cases where nodes (sensor locations, blogs) have different costs.

We evaluate our approach on several large real-world problems, including a model of a water distribution network from the EPA, and real blog data. The obtained sensor placements are provably near optimal, providing a constant fraction of the optimal solution. We show that the approach scales, achieving speedups and savings in storage of several orders of magnitude. We also show how the approach leads to deeper insights in both applications, answering multicriteria trade-off, cost-sensitivity and generalization questions.

8.1 Introduction

We explore the general problem of detecting outbreaks in networks, where we are given a network and a dynamic process spreading over this network, and we want to select a set of nodes to detect the process as effectively as possible.

Many real-world problems can be modeled under this setting. Consider a city water distribution network, delivering water to households via pipes and junctions. Accidental or malicious intrusions can cause contaminants to spread over the network, and we want to select a few locations (pipe junctions) to install sensors, in order to detect these contaminations as quickly as possible. In August 2006, the Battle of Water

Sensor Networks (BWSN) [Ostfeld et al., 2006] was organized as an international challenge to find the best sensor placements for a real (but anonymized) metropolitan area water distribution network. As part of this chapter, we present the approach we used in this competition. Typical epidemics scenarios also fit into this outbreak detection setting: We have a social network of interactions between people, and we want to select a small set of people to monitor, so that any disease outbreak can be detected early, when very few people are infected.

In the domain of weblogs (blogs), bloggers publish posts and use hyper-links to refer to other bloggers' posts and content on the web. Each post is time stamped, so we can observe the spread of information on the “blogosphere”. In this setting, we want to select a set of blogs to read (or retrieve) which are most up to date, *i.e.*, catch (link to) most of the stories that propagate over the blogosphere. Figure 8.1 illustrates this setting. Each layer plots the propagation graph (also called *information cascade* [Bikhchandani et al., 1992]) of the information. Circles correspond to blog posts, and all posts at the same vertical column belong to the same blog. Edges indicate the temporal flow of information: the cascade starts at some post (*e.g.*, top-left circle of the top layer of Figure 8.1) and then the information propagates recursively by other posts linking to it. Our goal is to select a small set of blogs (two in case of Figure 8.1) which “catch” as many cascades (stories) as possible¹. A naive, intuitive solution would be to select the big, well-known blogs. However, these usually have a large number of posts, and are time-consuming to read. We show, that, perhaps counterintuitively, a more cost-effective solution can be obtained, by reading smaller, but higher quality, blogs, which our algorithm can find.

There are several possible criteria one may want to optimize in outbreak detection. For example, one criterion seeks to minimize *detection time* (*i.e.*, to know about a cascade as soon as possible, or avoid spreading of contaminated water). Similarly, another criterion seeks to minimize the *population affected* by an undetected outbreak (*i.e.*, the number of blogs referring to the story we just missed, or the population consuming the contamination we cannot detect). Optimizing these objective functions is NP-hard, so for large, real-world problems, we cannot expect to find the optimal solution.

In this chapter, we show, that these and many other realistic outbreak detection objectives are *submodular*, *i.e.*, they exhibit a diminishing returns property: Reading a blog (or placing a sensor) when we have only read a few blogs provides more new information than reading it after we have read many blogs (placed many sensors).

We show how we can exploit this submodularity property to *efficiently obtain* solutions which are *provably close* to the optimal solution. These guarantees are important in practice, since selecting nodes is expensive (reading blogs is time-consuming, sensors have high cost), and we desire solutions which are not too far from the optimal solution.

The main contributions of this part of thesis are:

- We show that many objective functions for detecting outbreaks in networks are submodular, including detection time and population affected in the blogosphere and water distribution monitoring problems. We show that our approach also generalizes work by [Kempe et al., 2003] on selecting nodes maximizing influence in a social network.
- We exploit the submodularity of the objective (*e.g.*, detection time) to develop an efficient approximation algorithm, CELF, which achieves near-optimal placements (guaranteeing at least a constant fraction of the optimal solution), providing a novel theoretical result for non-constant node cost

¹In real-life multiple cascades can be on the same or similar story, but we still aim to detect as many as possible.

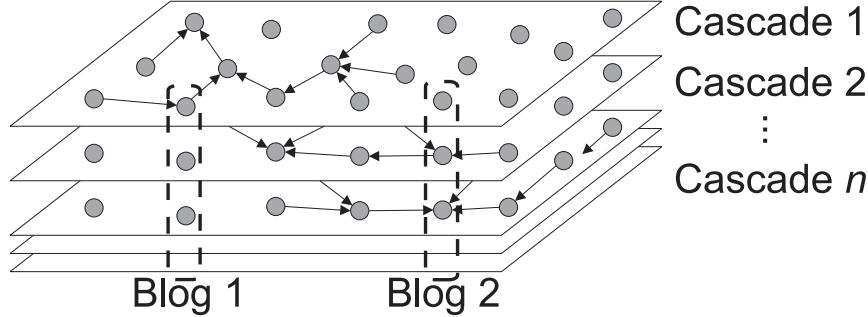


Figure 8.1: Spread of information between blogs. Each layer shows an information cascade, and all posts at the same vertical column belong to the same blog. Edges represent the flow of information. We want to pick a few blogs quickly capture most cascades.

SYMBOL	DESCRIPTION
\mathcal{G}	Graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
\mathcal{V}	Vertex set
\mathcal{E}	Edge set
$i \in \mathcal{I}$	Set of all possible outbreaks (set of all possible cascades)
$T(i, s)$	Time it takes an outbreak (cascade) i to reach node s
$c(s)$	Cost of monitoring (placing a sensor, reading a blog) s
$c(\mathcal{A})$	Cost of placement \mathcal{A} , $c(\mathcal{A}) = \sum_{s \in \mathcal{A}} c(s)$
$\pi(\mathcal{A})$	Expected penalty over all possible outbreaks \mathcal{I}
$R(\mathcal{A})$	Reward, i.e., penalty reduction $R(\mathcal{A}) = \pi(\emptyset) - \pi(\mathcal{A})$
δ_s	Marginal reward (gain), $\delta_s = R(\mathcal{A} \cup s) - R(\mathcal{A})$
s_k	Location with highest marginal reward or benefit/cost ratio

Table 8.1: Table of symbols.

functions. CELF is up to 700 times faster than simple greedy algorithm. We also derive novel online bounds on the quality of the placements obtained by *any* algorithm.

- We extensively evaluate our methodology on the applications introduced above – water quality and blogosphere monitoring. These are large real-world problems, involving a model of a water distribution network from the EPA with millions of contamination scenarios, and real blog data with millions of posts.
- We show how the proposed methodology leads to deeper insights in both applications, including multicriterion, cost-sensitivity analyses and generalization questions.

8.2 Outbreak Detection

8.2.1 Problem statement

The water distribution and blogosphere monitoring problems, even though in very different domains, share essential structure. In both problems, we want to select a subset \mathcal{A} of nodes (sensor locations, blogs) in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which detect outbreaks (spreading of a virus/information) quickly.

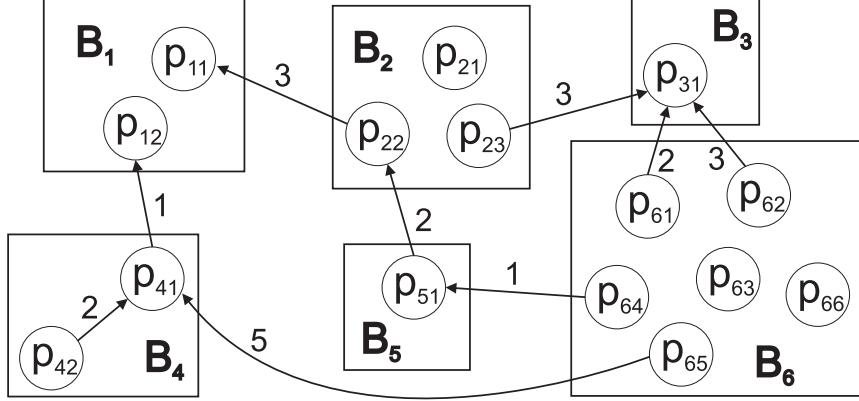


Figure 8.2: Blogs have posts, and there are time stamped links between the posts. The links point to the sources of information and the cascades grow (information spreads) in the reverse direction of the edges. Reading only blog B_6 captures all cascades, but late. B_6 also has many posts, so by reading B_1 and B_2 we detect cascades sooner.

Figure 8.2 presents an example of such a graph for blog network. Each of the six blogs consists of a set of posts. Connections between posts represent hyper-links, and labels show the time difference between the source and destination post, e.g., post p_{41} linked p_{12} one day after p_{12} was published).

These outbreaks (e.g., information cascades) initiate from a single node of the network (e.g., p_{11} , p_{12} and p_{31}), and spread over the graph, such that the traversal of every edge $(s, t) \in \mathcal{E}$ takes a certain amount of time (indicated by the edge labels). As soon as the event reaches selected node, alarm is triggered. E.g., selecting blog B_6 , would detect the cascades originating from post p_{11} , p_{12} and p_{31} , after 6, 6 and 2 timesteps after the start of the respective cascades.

Depending on which nodes we select, we achieve a certain placement score. Figure 8.2 illustrates several criteria one may want to optimize. If we only want to detect as many stories as possible, then reading just blog B_6 is best. However, reading B_1 would only miss one cascade (p_{31}), but would detect the other cascades immediately. In general, this placement score (representing, e.g., the fraction of detected cascades, or the population saved by placing a sensor) is a set function R , mapping every placement \mathcal{A} to a real number $R(\mathcal{A})$ (our reward), which we intend to maximize.

Since sensors are expensive, we also associate a *cost* $c(\mathcal{A})$ with every placement \mathcal{A} , and require, that this cost does not exceed a specified budget B which we can spend. For example, the cost of selecting a blog could be the number of posts in it (i.e., B_1 has cost 2, while B_6 has cost 6). In the water distribution setting, accessing certain locations in the network might be more difficult (expensive) than other locations. Also, we could have several types of sensors to choose from, which vary in their quality (detection accuracy) and cost. We associate a nonnegative cost $c(s)$ with every sensor s , and define the cost of placement \mathcal{A} : $c(\mathcal{A}) = \sum_{s \in \mathcal{A}} c(s)$.

Using this notion of reward and cost, our goal is to solve the optimization problem

$$\max_{\mathcal{A} \subseteq \mathcal{V}} R(\mathcal{A}) \text{ subject to } c(\mathcal{A}) \leq B, \quad (8.1)$$

where B is a budget we can spend for selecting the nodes.

8.2.2 Placement objectives

An event $i \in \mathcal{I}$ from set \mathcal{I} of scenarios (*e.g.*, cascades, contaminant introduction) originates from a node $s' \in \mathcal{V}$ of a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and spreads through the network, affecting other nodes (*e.g.*, through citations, or flow through pipes). Eventually, it reaches a monitored node $s \in \mathcal{A} \subseteq \mathcal{V}$ (*i.e.*, blogs we read, pipe junction we instrument with a sensor), and gets detected. Depending on the time of detection $t = T(i, s)$, and the impact on the network before the detection (*e.g.*, the size of the cascades missed, or the population affected by a contaminant), we incur *penalty* $\pi_i(t)$. Note that the penalty function $\pi_i(t)$ depends on the scenario. We discuss concrete examples of penalty functions below. Our goal is to minimize the expected penalty over all possible scenarios i :

$$\pi(\mathcal{A}) \equiv \sum_i P(i) \pi_i(T(i, \mathcal{A})),$$

where, for a placement $\mathcal{A} \subseteq \mathcal{V}$, $T(i, \mathcal{A}) = \min_{s \in \mathcal{A}} T(i, s)$ is the time until event i is detected by one of the sensors in \mathcal{A} , and P is a (given) probability distribution over the events.

We assume $\pi_i(t)$ to be monotonically nondecreasing in t , *i.e.*, we never prefer late detection if we can avoid it. We also set $T(i, \emptyset) = \infty$, and set $\pi_i(\infty)$ to some maximum penalty incurred for not detecting event i .

Proposed alternative formulation. Instead of minimizing the penalty $\pi(\mathcal{A})$, we can consider the scenario specific *penalty reduction* $R_i(\mathcal{A}) = \pi_i(\infty) - \pi_i(T(i, \mathcal{A}))$, and the expected penalty reduction

$$R(\mathcal{A}) = \sum_i P(i) R_i(\mathcal{A}) = \pi(\emptyset) - \pi(\mathcal{A}),$$

describes the expected benefit (reward) we get from placing the sensors. This alternative formulation has crucial properties which our method exploits, as described below.

Examples used in our experiments. Even though the water distribution and blogosphere monitoring problems are very different, similar placement objective scores make sense for both applications. The detection time $T(i, s)$ in the blog setting is the time difference in days, until blog s participates in the cascade i , which we extract from the data. In the water network, $T(i, s)$ is the time it takes for contaminated water to reach node s in scenario i (depending on outbreak location and time). In both applications we consider the following objective functions (penalty reductions):

- (a) *Detection likelihood (DL)*: fraction of information cascades and contamination events detected by the selected nodes. Here, the penalty is $\pi_i(t) = 0$, and $\pi_i(\infty) = 1$, *i.e.*, we do not incur any penalty if we detect the outbreak in finite time, otherwise we incur penalty 1.
- (b) *Detection time (DT)* measures the time passed from outbreak till detection by one of the selected nodes. Hence, $\pi_i(t) = \min\{t, T_{\max}\}$, where T_{\max} is the time horizon we consider (end of simulation / data set).
- (c) *Population affected (PA)* by scenario (cascade, outbreak). This criterion has different interpretations for both applications. In the blog setting, the affected population measures the number of blogs involved in a cascade before the detection. Here, $\pi_i(t)$ is the size of (number of blogs participating in) cascade i at time t , and $\pi_i(\infty)$ is the size of the cascade at the end of the data set. In the water distribution application, the affected population is the expected number of people affected by not (or late) detecting a contamination event.

Note, that optimizing each of the objectives can lead to very different solutions, hence we may want to simultaneously optimize all objectives at once. We deal with this multicriterion optimization problem in Section 8.2.4.

8.2.3 Properties of the placement objectives

The penalty reduction function² $R(\mathcal{A})$ has several important and intuitive properties: Firstly, $R(\emptyset) = 0$, *i.e.*, we do not reduce the penalty if we do not place any sensors. Secondly, R is nondecreasing, *i.e.*, $R(\mathcal{A}) \leq R(\mathcal{B})$ for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$. Hence, adding sensors can only decrease the incurred penalty. Thirdly, and most importantly, it satisfies the following intuitive diminishing returns property: If we add a sensor to a small placement \mathcal{A} , we improve our score at least as much, as if we add it to a larger placement $\mathcal{B} \supseteq \mathcal{A}$. More formally, we can prove that

Theorem 8.2.1. *For all placements $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$ and sensors $s \in \mathcal{V} \setminus \mathcal{B}$, it holds that*

$$R(\mathcal{A} \cup \{s\}) - R(\mathcal{A}) \geq R(\mathcal{B} \cup \{s\}) - R(\mathcal{B}).$$

A set function R with this property is called *submodular*.

Proof. Our proof is similar to the analysis of [Nemhauser et al., 1978]. Fix scenario i . We first show that the function $R_i(\mathcal{A}) = \pi_i(\infty) - \pi_i(T(\mathcal{A}, i))$ is submodular. Consider $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$. Let $s \in \mathcal{V} \setminus \mathcal{B}$. We have three cases. (i) $T(s, i) \geq T(\mathcal{A}, i)$. Then $T(\mathcal{A} \cup \{s\}) = T(\mathcal{A})$ and $T(\mathcal{B} \cup \{s\}) = T(\mathcal{B})$ and hence $R_i(\mathcal{A} \cup \{s\}) - R_i(\mathcal{A}) = 0 = R_i(\mathcal{B} \cup \{s\}) - R_i(\mathcal{B})$. (ii) $T(\mathcal{B}, i) \leq T(s, i) < T(\mathcal{A}, i)$. In this case, $R_i(\mathcal{A} \cup \{s\}) - R_i(\mathcal{A}) \geq 0 = R_i(\mathcal{B} \cup \{s\}) - R_i(\mathcal{B})$. Finally, (iii), $T(s, i) < T(\mathcal{B}, i)$. In this case, $R_i(\mathcal{A} \cup \{s\}) - R_i(\mathcal{A}) = [\pi_i(\infty) - \pi_i(T(s, i))] - R_i(\mathcal{A}) \geq [\pi_i(\infty) - \pi_i(T(s, i))] - R_i(\mathcal{B}) = R_i(\mathcal{B} \cup \{s\}) - R_i(\mathcal{B})$, where the inequality is due to the nondecreasingness of $R_i(\cdot)$. Hence, for each scenario i , the function R_i is submodular. Now, $R(\mathcal{A}) = \sum_i P(i)R_i(\mathcal{A})$ is a nonnegative linear combination of submodular functions, and hence submodular too. \square

Hence, both the blogosphere and water distribution monitoring problems can be reduced to the problem of maximizing a nondecreasing submodular function, subject to a constraint on the budget we can spend for selecting nodes. More generally, any objective function that can be viewed as an expected penalty reduction is submodular. Submodularity of R will be the key property exploited by our algorithms.

8.2.4 Multicriterion optimization

In practical applications, such as the blogosphere and water distribution monitoring, we may want to simultaneously optimize multiple objectives. Then, each placement has a vector of scores, $R(\mathcal{A}) = (R_1(\mathcal{A}), \dots, R_m(\mathcal{A}))$. Here, the situation can arise that two placements \mathcal{A}_1 and \mathcal{A}_2 are *incomparable*, *e.g.*, $R_1(\mathcal{A}_1) > R_1(\mathcal{A}_2)$, but $R_2(\mathcal{A}_1) < R_2(\mathcal{A}_2)$. So all we can hope for are *Pareto-optimal solutions* [Boyd and Vandenberghe, 2004]. A placement \mathcal{A} is called Pareto-optimal, if there does not exist another placement \mathcal{A}' such that $R_i(\mathcal{A}') \geq R_i(\mathcal{A})$ for all i , and $R_j(\mathcal{A}') > R_j(\mathcal{A})$ for some j (*i.e.*, there

²The objective R is similar to one of the examples of submodular functions described by [Nemhauser et al., 1978]. Our objective, however, preserves additional problem structure (sparsity) which we exploit in our implementation, and which we crucially depend on to solve large problem instances.

is no placement \mathcal{A}' which is at least as good as \mathcal{A} in all objectives R_i , and strictly better in at least one objective R_j .

One common approach for finding such Pareto-optimal solutions is by using *scalarization* (see for example, [Boyd and Vandenberghe, 2004]). Here, one picks positive weights $\lambda_1 > 0, \dots, \lambda_m > 0$, and optimizes the objective $R(\mathcal{A}) = \sum_i \lambda_i R_i(\mathcal{A})$. Any solution maximizing $R(\mathcal{A})$ is *guaranteed* to be Pareto-optimal [Boyd and Vandenberghe, 2004], and by varying the weights λ_i , different Pareto-optimal solutions can be obtained. One might be concerned that, even if optimizing the individual objectives R_i is easy (*i.e.*, can be approximated well), optimizing the sum $R = \sum_i \lambda_i R_i$ might be hard. However, submodularity is closed under nonnegative linear combinations and thus the new scalarized objective is submodular as well, and we can apply the algorithms we develop in the following section.

8.3 Proposed algorithm

Maximizing submodular functions in general is NP-hard [Khuller et al., 1999]. A commonly used heuristic in the *simpler* case, where every node has *equal* cost (*i.e.*, unit cost, $c(s) = 1$ for all locations s) is the *greedy algorithm*, which starts with the empty placement $\mathcal{A}_0 = \emptyset$, and iteratively, in step k , adds the location s_k which maximizes the *marginal gain*

$$s_k = \underset{s \in \mathcal{V} \setminus \mathcal{A}_{k-1}}{\operatorname{argmax}} R(\mathcal{A}_{k-1} \cup \{s\}) - R(\mathcal{A}_{k-1}). \quad (8.2)$$

The algorithm stops, once it has selected B elements. Considering the hardness of the problem, we might expect the greedy algorithm to perform arbitrarily badly. However, in the following we show that this is not the case.

8.3.1 Bounds for the algorithm

Unit cost case. Perhaps surprisingly – in the unit cost case – the simple greedy algorithm is near-optimal:

Theorem 8.3.1 ([Nemhauser et al., 1978]). *If R is a submodular, nondecreasing set function and $R(\emptyset) = 0$, then the greedy algorithm finds a set \mathcal{A}_G , such that $R(\mathcal{A}_G) \geq (1 - 1/e) \max_{|\mathcal{A}|=B} R(\mathcal{A})$.*

Hence, the greedy algorithm is guaranteed to find a solution which achieves at least a constant fraction $(1 - 1/e) \approx 63\%$ of the optimal score. The penalty reduction R satisfies all requirements of Theorem 8.3.1, and hence the greedy algorithm approximately solves the maximization problem Eq. (8.1).

Non-constant costs. What if the costs of the nodes are not constant? It is easy to see that the simple greedy algorithm, which iteratively adds sensors using rule from Eq. (8.2) until the budget is exhausted, can fail badly, since it is indifferent to the costs (*i.e.*, a very expensive sensor providing reward r is preferred over a cheaper sensor providing reward $r - \varepsilon$). To avoid this issue, the greedy rule Eq. (8.2) can be modified to take costs into account:

$$s_k = \underset{s \in \mathcal{V} \setminus \mathcal{A}_{k-1}}{\operatorname{argmax}} \frac{R(\mathcal{A}_{k-1} \cup \{s\}) - R(\mathcal{A}_{k-1})}{c(s)}, \quad (8.3)$$

i.e., the greedy algorithm picks the element maximizing the benefit/cost ratio. The algorithm stops once no element can be added to the current set \mathcal{A} without exceeding the budget. Unfortunately, this intuitive generalization of the greedy algorithm can perform arbitrarily worse than the optimal solution. Consider the case where we have two locations, s_1 and s_2 , $c(s_1) = \varepsilon$ and $c(s_2) = B$. Also assume we have only one scenario i , and $R(\{s_1\}) = 2\varepsilon$, and $R(\{s_2\}) = B$. Now, $R(\{s_1\}) - R(\emptyset)/c(s_1) = 2$, and $R(\{s_2\}) - R(\emptyset)/c(s_2) = 1$. Hence the greedy algorithm would pick s_1 . After selecting s_1 , we cannot afford s_2 anymore, and our total reward would be ε . However, the optimal solution would pick s_2 , achieving total penalty reduction of B . As ε goes to 0, the performance of the greedy algorithm becomes arbitrarily bad.

However, the greedy algorithm can be improved to achieve a constant factor approximation. This new algorithm, CEF (Cost-Effective Forward selection), computes the solution \mathcal{A}_{GCB} using the benefit-cost greedy algorithm, using rule (8.3), and also computes the solution \mathcal{A}_{GUC} using the unit-cost greedy algorithm (ignoring the costs), using rule (8.2). For both rules, CEF only considers elements which do not exceed the budget B . CEF then returns the solution with higher score. Even though both solutions can be arbitrarily bad, the following result shows that there is at least one of them which is not too far away from optimum, and hence CEF provides a constant factor approximation.

Theorem 8.3.2. *Let R be the a nondecreasing submodular function with $R(\emptyset) = 0$. Then*

$$\max\{R(\mathcal{A}_{GCB}), R(\mathcal{A}_{GUC})\} \geq \frac{1}{2}(1 - 1/e) \max_{\mathcal{A}, c(\mathcal{A}) \leq B} R(\mathcal{A}).$$

Proof. The proof is presented in our technical report [Krause and Guestrin, 2005] □

Theorem 8.3.2 was proved by [Khuller et al., 1999] for the special case of the Budgeted MAX-COVER problem³, and here we prove this result for *arbitrary* nondecreasing submodular functions. Theorem 8.3.2 states that the better solution of \mathcal{A}_{GBC} and \mathcal{A}_{GUC} (which is returned by CEF) is at most a constant factor $\frac{1}{2}(1 - 1/e)$ of the optimal solution.

Note that the running time of CEF is $\mathcal{O}(B|\mathcal{V}|)$ in the number of possible locations $|\mathcal{V}|$ (if we consider a function evaluation $R(\mathcal{A})$ as atomic operation, and the lowest cost of a node is constant). In [Sviridenko, 2004], it was shown that even in the non-constant cost case, the approximation guarantee of $(1 - 1/e)$ can be achieved. However, their algorithm is $\Omega(B|\mathcal{V}|^4)$ in the size of possible locations $|\mathcal{V}|$ we need to select from, which is prohibitive in the applications we consider. In the following, we show, that even the solutions of CEF are provably very close to the optimal score.

8.3.2 Online bounds for any algorithm

The approximation guarantees of $(1 - 1/e)$ and $\frac{1}{2}(1 - 1/e)$ in the unit- and non-constant cost cases are *offline*, *i.e.*, we can state them in advance before running the actual algorithm. We can also use submodularity to acquire tight *online* bounds on the performance of an *arbitrary* placement (not just the one obtained by the CEF algorithm).

Theorem 8.3.3. *For a placement $\widehat{\mathcal{A}} \subseteq \mathcal{V}$, and each $s \in \mathcal{V} \setminus \widehat{\mathcal{A}}$, let $\delta_s = R(\widehat{\mathcal{A}} \cup \{s\}) - R(\widehat{\mathcal{A}})$. Let $r_s = \delta_s/c(s)$, and let s_1, \dots, s_m be the sequence of locations with r_s in decreasing order. Let k be such*

³In MAX-COVER, we pick from a collection of sets, such that the union of the picked sets is as large as possible.

that $C = \sum_{i=1}^{k-1} c(s_i) \leq B$ and $\sum_{i=1}^k c(s_i) > B$. Let $\lambda = (B - C)/c(s_k)$. Then

$$\max_{\mathcal{A}, c(\mathcal{A}) \leq B} R(\mathcal{A}) \leq R(\widehat{\mathcal{A}}) + \sum_{i=1}^{k-1} \delta_{s_i} + \lambda \delta_{s_k}. \quad (8.4)$$

Proof. For all nodes $s \in \mathcal{V} \setminus \mathcal{A}$, let $\delta_s = R(\mathcal{A} \cup \{s\}) - R(\mathcal{A})$. Let us assume the costs $c(s)$ and B are rational. Without loss of generality, we can multiply costs and budget by their least common multiple, and hence we are left with integral costs and budget. Let us replicate all elements according to their cost, and assign weights to them according to their benefit cost ratio, *i.e.*, for all replica s' of element s , set weight $\delta'_{s'} = \delta_s / c(s)$. Also, let \mathcal{A}' be the set of all replicas corresponding to the nodes in \mathcal{A} . Let \mathcal{B}' be the replicas of all elements in the optimal solution \mathcal{B} . Since R is monotonic, $R(\mathcal{A}' \cup \mathcal{B}') \geq R(\mathcal{B}') = OPT$. Due to submodularity,

$$R(\mathcal{A}' \cup \mathcal{B}') \leq R(\mathcal{A}') + \sum_{s' \in \mathcal{B}'} \delta'_{s'}.$$

Furthermore,

$$\sum_{s' \in \mathcal{B}'} \delta'_{s'} \leq \max_{\mathcal{C}: |\mathcal{C}| \leq B} \sum_{s' \in \mathcal{C}} \delta'_{s'}.$$

Now we have a unit-cost modular optimization problem: We want to pick the best set \mathcal{C}' of B elements, maximizing the sum of their weights $\delta'_{s'}$. The ordinary unit cost greedy algorithm solves this optimally. More specifically, we can sort the new weights $\delta'_{s'}$ in decreasing order (in case of ties we keep the replicas of the elements in contiguous blocks), and pick the B first elements. Hence, the greedy algorithm on the replicated unit cost problem will have to integrally pick the first $k - 1$ original elements, and will fractionally pick the last $(k - th)$ element, selecting $M = (B - \sum_{i=1}^{k-1} c(s_i))$ elements. Summing up the weights of the unit cost elements will give us $\sum_i = 1^{k-1} \delta_{s_i} + \lambda * \delta_{s_k}$, where $\lambda = M/c(s_k)$. \square

Theorem 8.3.3 presents a way of computing how far any given solution $\widehat{\mathcal{A}}$ (obtained using CEF or *any* other algorithm) is from the optimal solution. This theorem can be readily turned into an algorithm, as formalized in Algorithm 2.

We empirically show that this bound is much tighter than the bound $\frac{1}{2}(1 - 1/e)$, which is roughly 31%.

8.4 Scaling up the algorithm

8.4.1 Speeding up function evaluations

Evaluating the penalty reductions R can be very expensive. For example, in the water distribution application, we need to run physical simulations, in order to estimate the effect of a contamination at a certain node. In the blog networks, we need to consider several millions of posts, which make up the cascades. However, in both applications, most outbreaks are sparse, *i.e.*, affect only a small area of the network (*c.f.*, [Krause et al., 2008, Leskovec et al., 2007d]), and hence are only detected by a small number of nodes. Hence, most nodes s do not reduce the penalty incurred by an outbreak (*i.e.*, $R_i(\{s\}) = 0$). Note, that this sparsity is *only* present if we consider penalty *reductions*. If for each sensor $s \in \mathcal{V}$ and scenario $i \in \mathcal{I}$ we store the actual penalty $\pi_i(s)$, the resulting representation is not sparse. Our implementation exploits

Function:LazyForward($\mathcal{G} = (\mathcal{V}, \mathcal{E}), R, c, B, type$)
 $\mathcal{A} \leftarrow \emptyset;$ **foreach** $s \in \mathcal{V}$ **do** $\delta_s \leftarrow +\infty;$
while $\exists s \in \mathcal{V} \setminus \mathcal{A} : c(\mathcal{A} \cup \{s\}) \leq B$ **do**
 foreach $s \in \mathcal{V} \setminus \mathcal{A}$ **do** $cur_s \leftarrow \text{false};$
 while true do
 if $type=UC$ **then** $s^* \leftarrow \underset{s \in \mathcal{V} \setminus \mathcal{A}, c(\mathcal{A} \cup \{s\}) \leq B}{\operatorname{argmax}} \delta_s;$
 if $type=CB$ **then** $s^* \leftarrow \underset{s \in \mathcal{V} \setminus \mathcal{A}, c(\mathcal{A} \cup \{s\}) \leq B}{\operatorname{argmax}} \frac{\delta_s}{c(s)};$
 if cur_{s^*} **then** $\mathcal{A} \leftarrow \mathcal{A} \cup \{s^*\}; \text{break};$
 else $\delta_{s^*} \leftarrow R(\mathcal{A} \cup \{s^*\}) - R(\mathcal{A}); cur_{s^*} \leftarrow \text{true};$
 end
end
return $\mathcal{A};$

Algorithm 8.1: The CELF algorithm.

Algorithm:CELF($\mathcal{G} = (\mathcal{V}, \mathcal{E}), R, c, B$)
 $\mathcal{A}_{UC} \leftarrow \text{LazyForward}(\mathcal{G}, R, c, B, UC);$
 $\mathcal{A}_{CB} \leftarrow \text{LazyForward}(\mathcal{G}, R, c, B, CB);$
return $\operatorname{argmax}\{R(\mathcal{A}_{UC}), R(\mathcal{A}_{CB})\}$

this sparsity by representing the penalty function R as an *inverted index*⁴, which allows fast lookup of the penalty reductions *by sensor index* s . By looking up all scenarios detected by all sensors in our placement \mathcal{A} , we can quickly compute the penalty reduction

$$R(\mathcal{A}) = \sum_{i: i \text{ detected by } \mathcal{A}} P(i) \max_{s \in \mathcal{A}} R_i(\{s\}),$$

without having to scan the entire data set.

The inverted index is the main data structure we use in our optimization algorithms. After the problem (water distribution network simulations, blog cascades) has been compressed into this structure, we use the same implementation for optimizing sensor placements and computing bounds.

In the water distribution network application for example, exploiting this sparsity allows us to fit the set of all possible intrusions considered in the BWSN challenge in main memory (16 GB), which leads to several orders of magnitude improvements in the running time, since we can avoid hard-drive accesses.

8.4.2 Reducing function evaluations

Even if we can quickly evaluate the score $R(\mathcal{A})$ of any given placement, we still need to perform a large number of these evaluations in order to run the greedy algorithm. If we select k sensors among $|\mathcal{V}|$ locations, we roughly need $k|\mathcal{V}|$ function evaluations. We can exploit submodularity further to require

⁴The index is inverted, since the data set facilitates the lookup *by scenario index* i (since we need to consider cascades, or contamination simulations for each scenario).

Algorithm 8.2: Getting bound \hat{R} on optimal solution.

Algorithm: GetBound($\mathcal{G} = (\mathcal{V}, \mathcal{E}), \mathcal{A}, R, c, B$)
 $\mathcal{A} \leftarrow \emptyset; \mathcal{B} \leftarrow \emptyset; \hat{R} = R(\mathcal{A})$;
foreach $s \in \mathcal{V}$ **do** $\delta_s \leftarrow R(\mathcal{A} \cup \{s\}) - R(\mathcal{A}); r_s = \frac{\delta_s}{c(s)}$;
while $\exists s \in \mathcal{V} \setminus (\mathcal{A} \cup \mathcal{B}) : c(\mathcal{A} \cup \mathcal{B} \cup \{s\}) \leq B$ **do**
 $s^* \leftarrow \operatorname{argmax}_{s \in \mathcal{V} \setminus \{\mathcal{A} \cup \mathcal{B}\}, c(\mathcal{A} \cup \mathcal{B} \cup \{s\}) \leq B} r_s$;
 $\hat{R} \leftarrow \hat{R} + \delta_{s^*}; \mathcal{B} \leftarrow \mathcal{B} \cup \{s^*\}$;
end
 $s^* \leftarrow \operatorname{argmax}_{s \in \mathcal{V} \setminus \{\mathcal{A} \cup \mathcal{B}\}, c(\mathcal{A} \cup \mathcal{B} \cup \{s\}) \leq B} r_s; \lambda \leftarrow \frac{B - c(\mathcal{A} \cup \mathcal{B})}{c(s^*)}$;
return $\hat{R} + \lambda \delta_{s^*}$;

far fewer function evaluations in practice. Assume we have computed the marginal increments $\delta_s(\mathcal{A}) = R(\mathcal{A} \cup \{s\}) - R(\mathcal{A})$ (or $\delta_s(\mathcal{A})/c(s)$) for all $s \in \mathcal{V} \setminus \mathcal{A}$. The key idea is to realize that, as our node selection \mathcal{A} grows, the marginal increments $\delta_{s'}$ (and $\delta_{s'}/c(s')$) (*i.e.*, the benefits for adding sensor s') can never increase: For $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, it holds that $\delta_s(\mathcal{A}) \geq \delta_s(\mathcal{B})$. So instead of recomputing $\delta_s \equiv \delta_s(\mathcal{A})$ for every sensor after adding s' (and hence requiring $|\mathcal{V}| - |\mathcal{A}|$ evaluations of R), we perform *lazy* evaluations: Initially, we mark all δ_s as *invalid*. When finding the next location to place a sensor, we go through the nodes in decreasing order of their δ_s . If the δ_s for the top node s is invalid, we recompute it, and insert it into the existing order of the δ_s (*e.g.*, by using a priority queue). In many cases, the recomputation of δ_s will lead to a new value which is not much smaller, and hence often, the top element will stay the top element even after recomputation. In this case, we found a new sensor to add, without having reevaluated δ_s for every location s . The correctness of this lazy procedure follows directly from submodularity, and leads to far fewer (expensive) evaluations of R . We call this lazy greedy algorithm⁵ CELF (Cost-Effective Lazy Forward selection). In our experiments, CELF achieved up to a factor 700 improvement in speed compared to CEF when selecting 100 blogs. Algorithm 8.1 provides pseudo-code for an implementation of CELF.

When computing the online bounds discussed in Section 8.3.2, we can use a similar lazy strategy. The only difference is that, instead of lazily ensuring that the best δ_s is correctly computed, we ensure that the top k (where k is as in Eq. (8.4)) δ_s improvements have been updated.

8.5 Case study: Blog Network

We begin by describing the blog network dataset, experimental setup, and objective function we consider. We then present results on solution quality, scalability and generalization to future data. We also explore various ways of assigning costs to blogs.

⁵[Robertazzi and Schwartz, 1989] suggested a similar algorithm for the *unit cost* case.

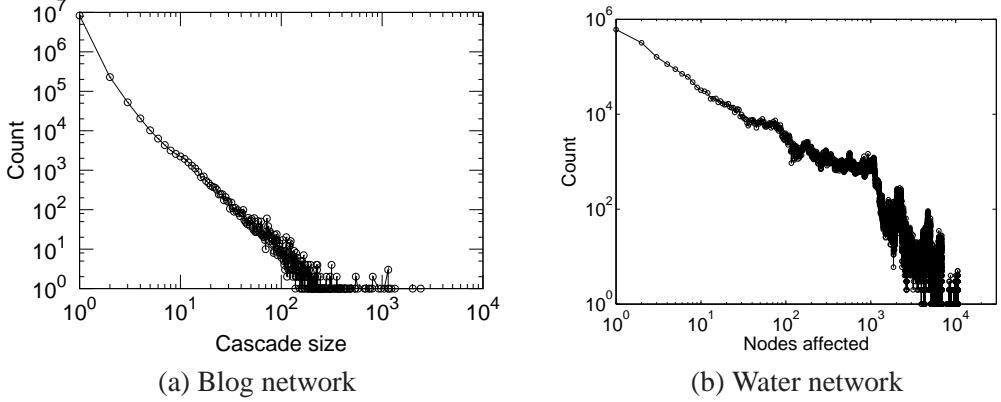


Figure 8.3: Cascade and outbreak size distributions for blog network and the water distribution network.

8.5.1 Experimental setup

In this work we are not explicitly modeling the spread of information over the network, but rather consider cascades as *input* to our algorithms.

Here we are interested in blogs that actively participate in discussions, we biased the dataset towards the active part of the blogosphere, and selected a subset from the larger set of 2.5 million blogs of [Glance et al., 2005]. We considered all blogs that received at least 3 in-links in the first 6 months of 2006, and then took all their posts for the full year 2006. So, the dataset that we use has 45,000 blogs, 10.5 million posts, and 16.2 million links (30 GB of data). However, only 1 million links point inside the set of 45,000 blogs.

Posts have rich metadata, including time stamps, which allows us to extract information cascades, *i.e.*, subgraphs induced by directed edges representing the temporal flow of information. We adopt the following definition of a cascade [Leskovec et al., 2007d]: every cascade has a single starting post, and other posts recursively join by linking to posts within the cascade, whereby the links obey time order. We detect cascades by first identifying starting post and then following in-links. We discover 346,209 non-trivial cascades having at least 2 nodes. Since the cascade size distribution is heavy-tailed, we further limit our analysis to only cascades that had at least 10 nodes. The final dataset has 17,589 cascades, where each blog participates in 9.4 different cascades on average.

8.5.2 Objective functions

We use the penalty reduction objectives DL, DT and PA as introduced in Section 8.2.2. We normalize the scores of the solution to be between 0 and 1. For the DL (detection likelihood) criterion, the quality of the solution is the fraction of all detected cascades (regardless of when we detect it). The PA (population affected) criterion measures what fraction of the population included in the cascade after we detect it, *i.e.*, if we would be reading all the blogs initiating the cascades, then the quality of the solution is 1. In PA our reward depends on which fraction of the cascades we detect, and big cascades count more than small cascades.

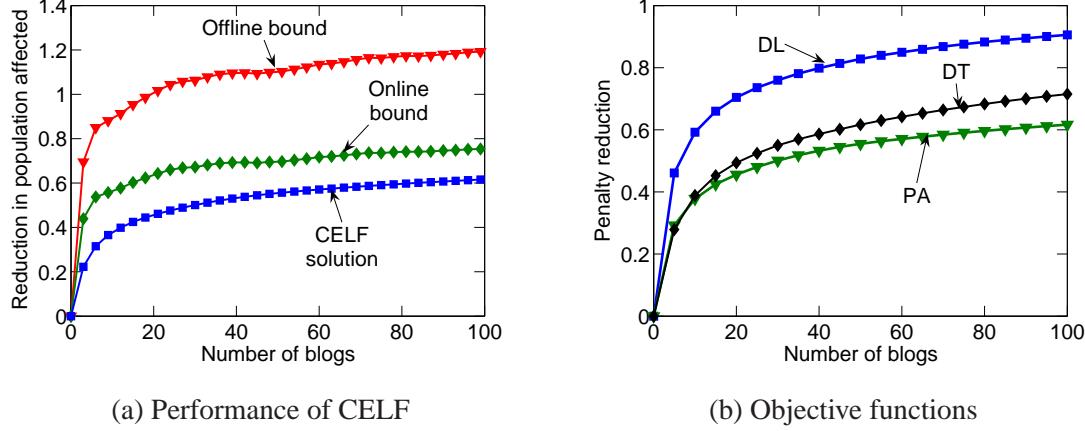


Figure 8.4: (a) Performance of CELF algorithm and off-line and on-line bounds for PA objective function.
(b) Compares objective functions.

8.5.3 Solution quality

First, we evaluate the performance of CELF, and estimate how far from optimal the solution could be. Note, that obtaining the optimal solution would require enumeration of $2^{45,000}$ subsets. Since this is impractical, we compare our algorithm to the bounds we developed in Section 8.3. Figure 8.4(a) shows scores for increasing budgets when optimized the PA (population affected) criterion. As we select more blogs to read, the proportion of cascades we catch increases (bottom line). We also plot the two bounds. The off-line bound (Section 8.3.1) shows that the unknown optimal solution lies between our solution (bottom line) and the bound (top line). Notice the discrepancy between the lines is big, which means the bound is very loose. On the other hand, the middle line shows the online bound (Section 8.3.2), which again tells us that the optimal solution is somewhere between our current solution and the bound. Notice, the gap is much smaller. This means (a) that the our on-line bound is much tighter than the traditional off-line bound. And, (b) that our CELF algorithm performs very close to the optimum.

In contrast to off-line bound, the on-line bound is *algorithm independent*, and thus can be computed regardless of the algorithm used to obtain the solution. Since it is tighter, it gives a much better worst case estimate of the solution quality. For this particular experiment, we see that CELF works very well: after selecting 100 blogs, we are at most **13.8%** away from the optimal solution.

Figure 8.4(b) shows the performance using various objective functions (from top to bottom: DL, DT, PA). DL increases the fastest, which means that one only needs to read a few blogs to detect most of the cascades, or equivalently that most cascades hit one of the big blogs. However, the population affected (PA) increases much slower, which means that one needs many more blogs to know about stories before the rest of population does. By using the on-line bound we also calculated that all objective functions are at most 5% to 15% from optimal.

8.5.4 Cost of a blog

The results presented so far assume that every blog has the same cost. Under this *unit cost* model, the algorithm tends to pick large, influential blogs, that have many posts. For example, `instapundit.com`

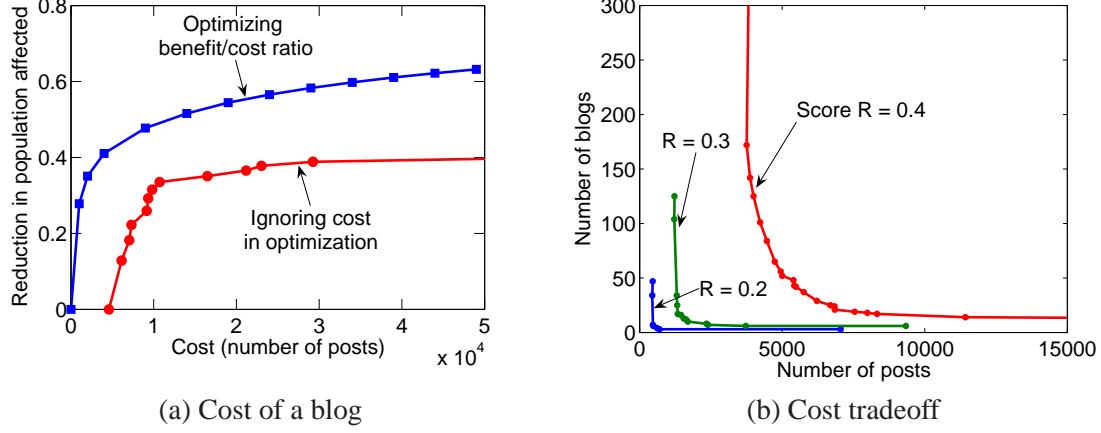


Figure 8.5: (a) Comparison of the unit and the number of posts cost models. (b) For fixed value of PA R , we get multiple solutions varying in costs.

is the best blog when optimizing PA, but it has 4,593 posts. Interestingly, most of the blogs among the top 10 are politics blogs like: instapundit.com, blogometer.nationaljournal.com, michellemalkin.com, and sciencepolitics.blogspot.com. Some popular aggregators of interesting things on the blogosphere are also selected: boingboing.net, themodulator.org and bloggersblog.com. The top 10 PA blogs had more than 21,000 thousand posts in 2006. They account for 0.2% of all posts, 3.5% of all in-links, 1.7% of out-links inside the dataset, and 0.37% of all out-links.

Under unit cost model large blogs are important, but reading a blog with many posts is time consuming. This motivates the *number of posts* (NP) cost model, where we set the cost of a blog to the number of posts it had in 2006.

First, we compare the NP cost model with the unit cost in Figure 8.5(a). The top curve shows the value of the PA criterion for budgets of B posts, *i.e.*, we optimize PA such that the selected blogs can have at most B posts total. Note, that under the unit cost model, CELF chooses expensive blogs with many posts. For example, to obtain the same PA objective value, one needs to read 10,710 posts under unit cost model. The NP cost model achieves the same score while reading just 1,500 posts. Thus, optimizing the benefit cost ratio (PA/cost) leads to drastically improved performance.

Interestingly, the solutions obtained under the NP cost model are very different from the unit cost model. Under NP , political blogs are not chosen anymore, but rather summarizers (*e.g.*, themodulator.org, watcherofweasels.com, anglican.tk) are important. Blogs selected under NP cost appear about 3 days later in the cascade as those selected under unit cost, which further suggests that summarizer blogs tend to be chosen under NP model.

In practice, the cost of reading a blog is not simply proportional to the number of posts, since we also need to navigate to the blog (which takes constant effort per blog). Hence, a combination of unit and NP cost is more realistic. Figure 8.5(b) interpolates between these two cost models. Each curve shows the solutions with the same value R of the PA objective, but using a different number of posts (x-axis) and blogs (y-axis) each. For a given R , the ideal spot is the one closest to origin, which means that we want to read the least number of posts from least blogs to obtain desired score R . Only at the end points does CELF tend to pick extreme solutions: few blogs with many posts, or many blogs with few posts. Note,

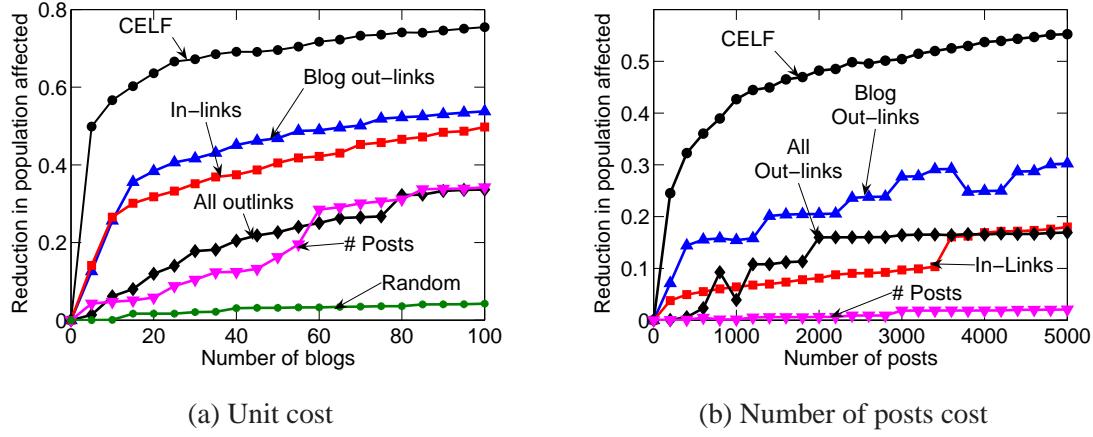


Figure 8.6: Heuristic blog selection methods. (a) unit cost model, (b) number of posts cost model.

there is a clear knee on plots of Figure 8.5(b), which means that by only slightly increasing the number of blogs we allow ourselves to read, the number of posts needed decreases drastically, while still maintaining the same value R of the objective function.

8.5.5 Comparison to heuristic blog selection

Next, we compare our method with several intuitive heuristic selection techniques. For example, instead of optimizing the DT, DL or PA objective function using CELF, we may just want to select the most popular blogs and hope to detect many cascades. We considered several such heuristics, where we order blogs by some “goodness” criteria, and then pick top blogs (until the budget is exhausted). We consider the following criteria: the number posts on the blog, the cumulative number of out-links of blog’s posts, the number of in-links the blog received from other blogs in the dataset, and the number of out-links to other blogs in the dataset.

As Figure 8.6(a) shows, the CELF algorithm greatly outperforms all the heuristic selection techniques. More interestingly, the best heuristics (doing 45% worse than CELF) pick blogs by the number of in- or out-links from/to other blogs in the dataset. Number of posts, the total number of out-links and random blog selection do not perform well.

Number of in-links is the indicator of a blog’s tendency to create cascades, while number of out-links (to other blogs) indicates blog’s tendency to summarize the blogosphere. We also note, that the surprisingly good performance of the number of out-links to blogs in the dataset is an artefact of our “closed-world” dataset, and in real-life we can not estimate this. The results also agree well with our intuition that the number of in-links is a good heuristic, since it directly indicates the of propagation of information.

Figure 8.6(b) explores the same setting under the NP cost model. Here, given a budget of B posts, we select a set of blogs to optimize PA objective. For the heuristics, we select a set of blogs to optimize chosen heuristic, *e.g.*, the total number of in-links of selected blogs while still fitting inside the budget of B posts. Again, CELF outperforms the next best heuristics by 41%, and again the number of in- and out-links are the best heuristics.

These results show that simple heuristics that one could use to identify blogs to read do not really work

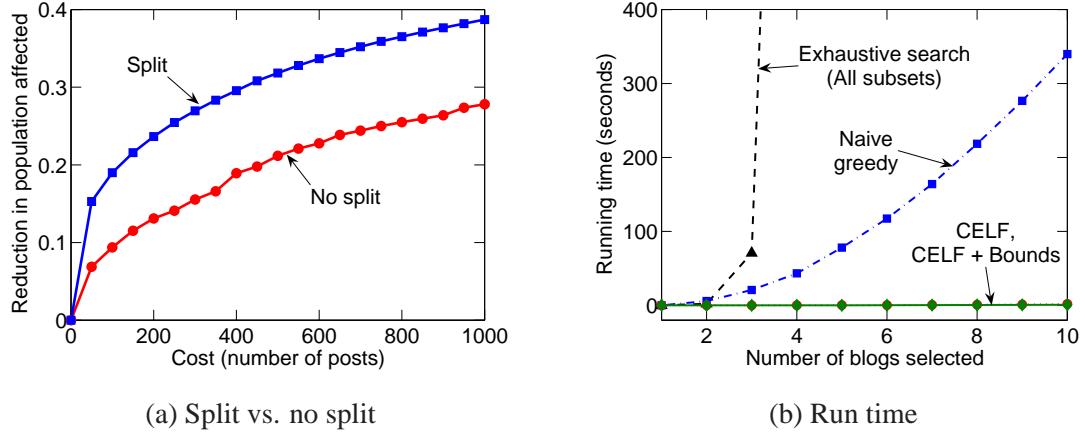


Figure 8.7: (a) Improvement in performance by splitting big blogs into multiple nodes. (b) Run times of exhaustive search, greedy and CELF algorithm.

well. There are good summarizer blogs that may not be very popular, but which, by using few posts, catch most of the important stories propagating over the blogosphere.

8.5.6 Fractionally selecting blogs

Our framework also allows fractional selection of blogs, which means that instead of reading a large blog every day, we can read it, *e.g.*, only one day per week. This also allows us to ask: what is the best day of the week to read blogs?

In order to study whether fractional selection allows to achieve better benefit cost ratio, we split the blogs which had at least one post per day into 7 blogs, one for each day of the week. Figure 8.7(a) shows, that by splitting big blogs, the population affected (PA) objective function increases for 12% over the setting where only whole blogs can be selected.

Returning to the original question, we performed the following experiment: given a budget of 1000 posts, what is the best day of the week to read posts (optimizing PA)? We found that Friday is the best day to read blogs. The value of PA for Friday is 0.20, while it is 0.13 for the rest of the week. We consider this surprising, since the activity of the blogosphere (number of posts and links created) drops towards the end of the week, and especially over the weekend [Leskovec et al., 2007d].

8.5.7 Generalization to future data

Since the influence and popularity of the blogs also evolves over time we also want to know how well the selected blogs will detect cascades in the future. To evaluate the generalization to unknown future, we use the first 6 months of the dataset as historic data to select a set of blogs, and then use second 6 months of the dataset to evaluate the performance of selected blogs on unseen future cascades.

Figure 8.8 compares the performance on the unknown future data. Top dashed curve in both plots shows the optimal performance on future data, *i.e.*, we select the blogs directly using the (unknown) future data.

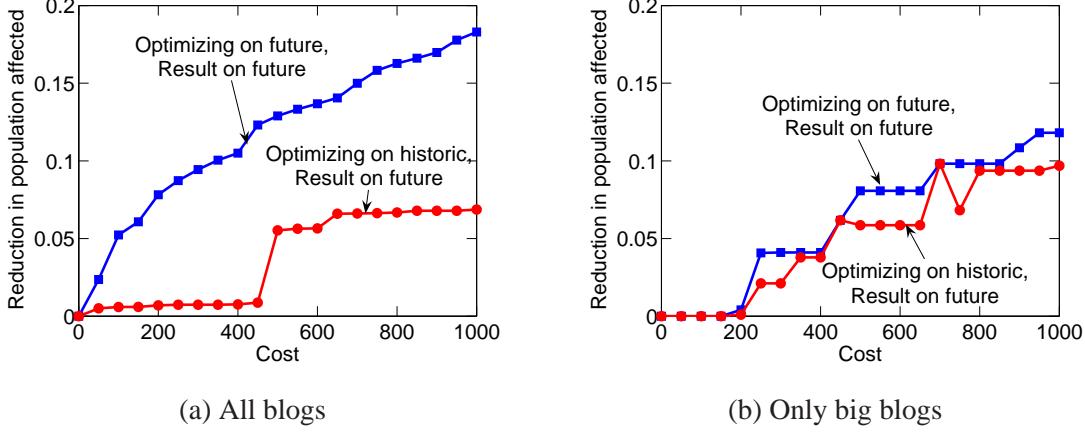


Figure 8.8: Generalization to future data when CELF can select any blog (a), or only big blogs (b).

The bottom curve presents the realistic case where we select the blogs using historic data and evaluate using hidden future data.

As Figure 8.8(a) shows, CELF overfits when evaluated on the future data, *i.e.*, it selects small blogs with very few posts that just by chance participate in cascades, and then these blogs do not generalize well for the second half of the year. One way to overcome this overfitting is to prevent CELF from picking very small blogs. To understand this restriction we show in Figure 8.8(b) the performance when CELF can only select blogs with at least one post per day (365 posts per year).

Comparing Figure 8.8(a) and Figure 8.8(b) we see that the optimal performance (top curve) drops if CELF is limited on only picking big blogs. This is expected since CELF has less choice of which blogs to pick, and thus performs worse. However, when limiting the selection to only big blogs (Figure 8.8(b)) the gap between the curves is very small (compared to the big gap of Figure 8.8(a)). Moreover, the performance on the future data does not drop, and the method generalizes well.

8.5.8 Scalability

Figure 8.5(b) plots the running time of selecting k blogs. We see that exhaustively enumerating all possible subsets of k elements is infeasible (the line jumps out of the plot for $k = 3$). The simple greedy algorithm scales as $\Omega(k|\mathcal{V}|)$, since for every increment of k we need to consider selecting all remaining $|\mathcal{V}|-k$ blogs. The bottom line overlapping the x-axis of Figure 8.5(b) shows the performance of our CELF algorithm. For example, for selecting 100 blogs, greedy algorithm runs 4.5h, while CELF takes 23 seconds (700 times faster). Calculation of the on-line bounds while running CELF takes 54s.

Exploiting the sparsity of the problem (*c.f.*, Section 8.4) allowed us to reduce the size of the inverted index from originally 3.5 GB to 50 MB, easily fitting it in main memory.

8.6 Case study: Water networks

Next we present our results on water distribution networks, where instead of information cascades the task is to detect contamination cascades.

8.6.1 Experimental setup

In the water distribution system application, we used the data and rules introduced by the Battle of Water Sensor Networks (BWSN) challenge [Ostfeld et al., 2006]. We considered both the small network on 129 nodes (BWSN1), and a large, realistic, 12,527 node distribution network (BWSN2) provided as part of the BWSN challenge. In addition we also consider a third water distribution network (NW3) of a large metropolitan area in the United States. The network (not including the household level) contains 21,000 nodes and 25,000 pipes (edges). To our knowledge, this is the largest water distribution network considered for sensor placement optimization so far. The networks consist of a static description (junctions and pipes) and dynamic parameters (time-varying water consumption demand patterns at different nodes, opening and closing of valves, pumps, tanks, etc.)

As Figure 8.3(b) shows, the distribution of outbreak sizes for the water network is rather different than for a blog network. The blog network is a typical scale free network with small diameter and power law degree distribution. On the other hand, the water networks are composed of several connected grid networks corresponding to different neighborhoods, and thus the outbreak size distribution is different.

8.6.2 Objective functions

In the BWSN challenge, we want to select a set of 20 sensors, simultaneously optimizing the objective functions DT, PA and DL, as introduced in Section 8.2.2. To obtain cascades we use a realistic disease model defined by [Ostfeld et al., 2006], which depends on the demands and the contaminant concentration at each node. In order to evaluate these objectives, we use the EPANET simulator [Rossman, 1999], which is based on a physical model to provide realistic predictions on the detection time and concentration of contaminant for any possible contamination event. We consider simulations of 48 hours length, with 5 minute simulation timesteps. Contaminations can happen at any node and any time within the first 24 hours, and spread through the network according to the EPANET simulation. The time of the outbreak is important, since water consumption varies over the day and the contamination spreads at different rates depending on the time of the day. Altogether, we consider a set of 3.6 million possible contamination scenarios and each of these is associated with a “cascade” of contaminant spreading over the network.

8.6.3 Solution quality

We first used CELF to optimize placements of increasing size, according to the three criteria DL, DT, PA. We again normalized the scores to be between 0 and 1, where 1 is the best achievable score when placing sensors at every node.

Figure 8.9 (a) presents the CELF score, the off-line and on-line bounds for PA objective on the BWSN2 network. Consistently with the blog experiments, the on-line bound is much tighter than the off-line bound, and the solutions obtained by our CELF algorithm are very close to the optimum.

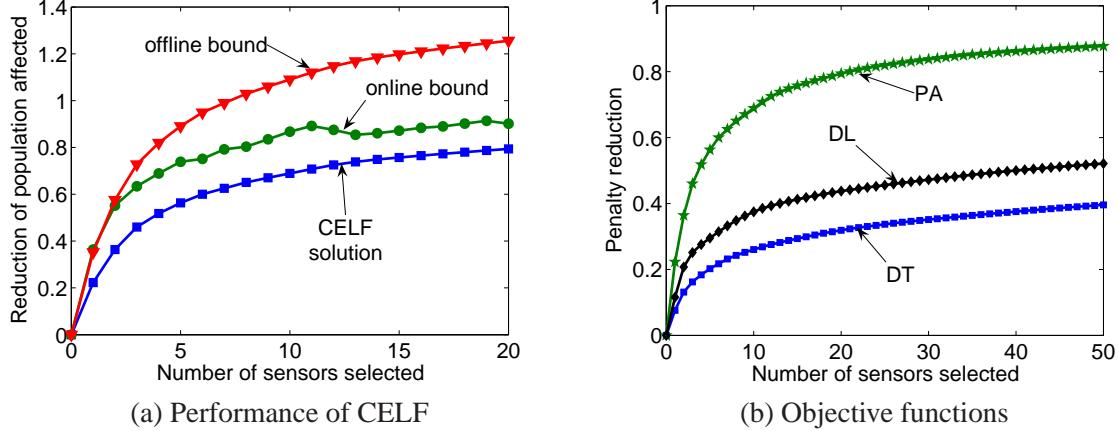


Figure 8.9: (a) CELF with offline and online bounds for PA objective. (b) Different objective functions.

Figure 8.9 (b) shows CELF’s performance on all 3 objective functions. Similarly to the blog data, the population affected (PA) score increases very quickly. The reason is that most contamination events only impact a small fraction of the network. Using few sensors, it is relatively easy to detect most of the high impact outbreaks. However, if we want to detect all scenarios, we need to place a large number of sensors (2,263 in our experiment). Hence, the DL (and correspondingly DT) increase more slowly than PA.

Figure 8.10 shows two 20 sensor placements after optimizing DL and PA respectively on BWSN2. When optimizing the population affected (PA), the placed sensors are concentrated in the dense high-population areas, since the goal is to detect outbreaks which affect the population the most. When optimizing the detection likelihood, the sensors are uniformly spread out over the network. Intuitively this makes sense, since according to BWSN challenge [Ostfeld et al., 2006], outbreaks happen with same probability at every node. So, for DL, the placed sensors should be as close to all nodes as possible.

We also compared the scores achieved by CELF with several heuristic sensor placement techniques, where we order nodes by some “goodness” criteria, and then pick top nodes. We consider the following criteria: population at the node, water flow through the node, and the diameter and the number of pipes at the node. Figure 8.13(a) shows the results for PA objective function. CELF outperforms best heuristic for 45%. Best heuristics are placing nodes at random, by degree or their population. We see heuristics perform poorly, since nodes which are close in the graph tend to have similar flow, diameter and population, and hence the sensors will be spread out too little. Even the maximum over one hundred random trials performs far worse than CELF. Figure 8.11(a) shows the statistics of choosing 100 random placements on the water distribution network for the PA objective function. Notice that even best out of 100 random trials performs far worse than CELF. Figure 8.11(b) shows how many outbreaks one needs so that the score approaches the true score that one obtains if data on all outbreaks is available. Notice that estimates soon converge to true score and data on less than 100,000 outbreaks is needed. See [Krause et al., 2008] for more details.

8.6.4 Multicriterion optimization

Using the theory developed in Section 8.2.4, we traded-off different objectives for the water distribution application. We selected pairs of objectives, *e.g.*, DL and PA, and varied the weights λ to produce (approx-

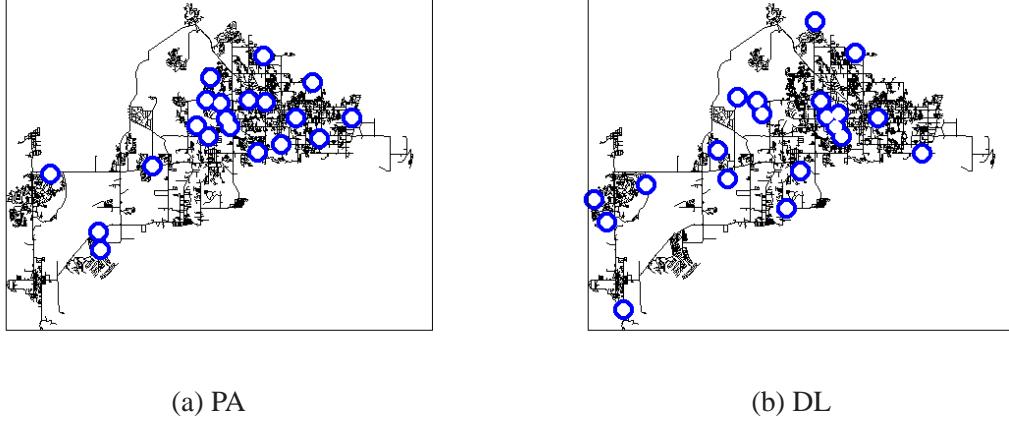


Figure 8.10: Water network sensor placements: (a) when optimizing PA, sensors are concentrated in high population areas. (b) when optimizing DL, sensors are uniformly spread out.

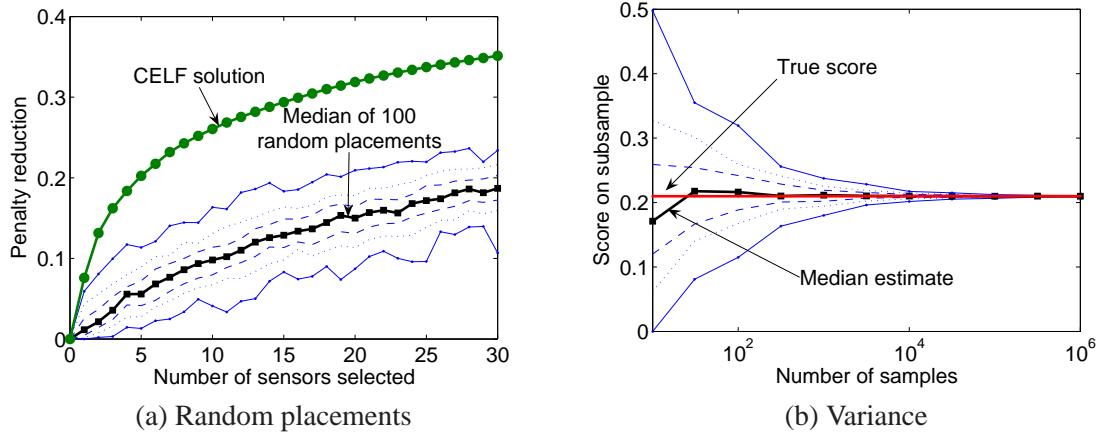


Figure 8.11: (a) Performance of 100 random placements on the water distribution network for the PA objective function. (b) Performance with the number of simulated outbreaks. As more outbreaks are available we get better performance.

imately) Pareto-optimal solutions. In Figure 8.12 (a) we plot the tradeoff curves for different placement sizes k . By adding more sensors, both objectives DL and PA increase. The curves also show, that if we, e.g., optimize for DL, the PA score can be very low. However, there are points which achieve near-optimal scores in both criteria (the *knee* in the curve). This sweet spot is what we aim for in multi-criteria optimization.

We also traded off the affected population PA and a fourth criterion defined by BWSN, the *expected consumption of contaminated water*. Figure 8.12 (b) shows the trade-off curve for this experiment. Notice that the curves (almost) collapse to points, indicating that these criteria are highly correlated, which we expect for this pair of objective functions. Again, the efficiency of our implementation allows to quickly generate and explore these trade-off curves, while maintaining strong guarantees about near-optimality of the results.

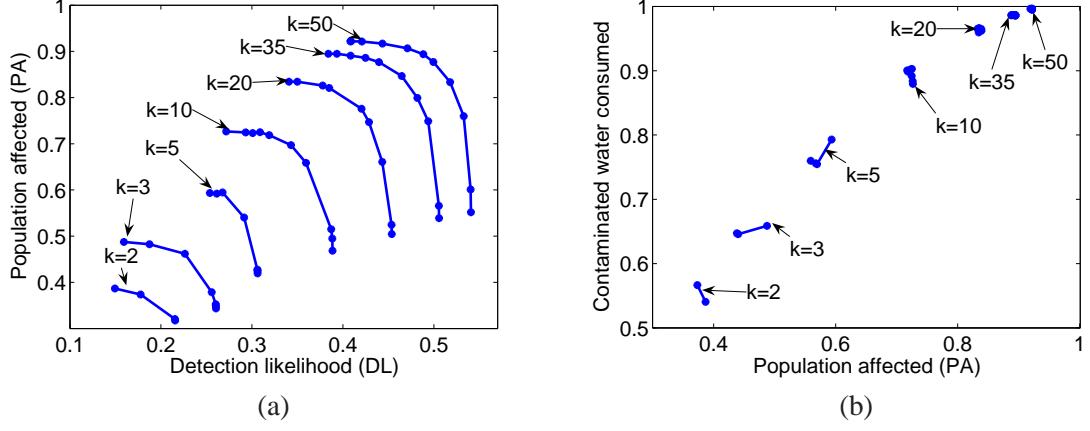


Figure 8.12: (a) Trading off PA and DL. (b) Trading off PA and consumed contaminated water.

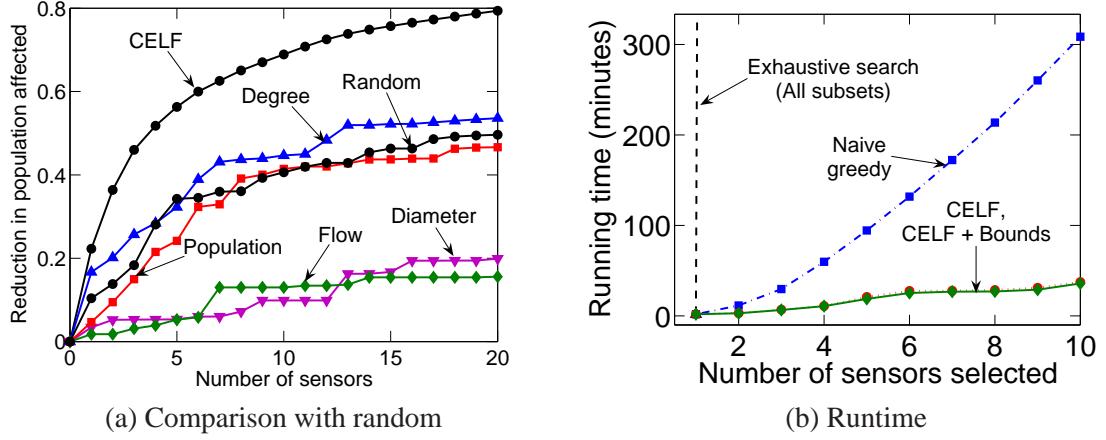


Figure 8.13: (a) Solutions of CELF outperform heuristic selections. (b) Running time of exhaustive search, greedy and CELF.

8.6.5 Scalability

In the water distribution setting, we need to simulate 3.6 million contamination scenarios, each of which takes approximately 7 seconds and produces 14KB of data. Since most of the computer cluster scheduling systems break if one would submit 3.6 million jobs into the queue, we developed a distributed architecture, where the clients obtain simulation parameters and then confirm the successful completion of the simulation. We run the simulation for a month on a cluster of around 40 machines. This produced 152GB of outbreak simulation data. By exploiting the properties of the problem described in Section 8.4, the size of the inverted index (which represents the relevant information for evaluating placement scores) is reduced to 16 GB which we were able to fit into main memory of a server. The fact that we could fit the data into main memory alone sped up the algorithms by at least a factor of 1000.

Figure 8.13 (b) presents the running times of CELF, the naive greedy algorithm and exhaustive search (extrapolated). We can see that the CELF is 10 times faster than the greedy algorithm when placing 10 sensors. Again, a drastic speedup.

8.7 Discussion and connection to previous work

Next we briefly discuss connections to previous work on influence maximization, optimization of submodular functions and modeling cascading behaviors in general.

8.7.1 Relationship to Influence Maximization

In [Kempe et al., 2003], a *Triggering Model* was introduced for modeling the spread of influence in a social network. As the authors show, this model generalizes the Independent Cascade, Linear Threshold and Listen-once models commonly used for modeling the spread of influence. Essentially, this model describes a probability distribution over directed graphs, and the influence is defined as the expected number of nodes reachable from a set of nodes, with respect to this distribution. Kempe et al. showed that the problem of selecting a set of nodes with maximum influence is submodular, satisfying the conditions of Theorem 8.3.1, and hence the greedy algorithm provides a $(1 - 1/e)$ approximation. The problem addressed in this chapter generalizes this Triggering model:

Theorem 8.7.1. *The Triggering Model [Kempe et al., 2003] is a special case of our network outbreak detection problem.*

In order to prove Theorem 8.7.1, we consider fixed directed graphs sampled from the Triggering distribution. If we revert the arcs in any such graph, then our PA objective corresponds exactly to the influence function of [Kempe et al., 2003] applied to the original graph.

Proof. Let P be a distribution over directed graphs $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1), \dots, \mathcal{G}_N = (\mathcal{V}, \mathcal{E}_N)$ on a fixed set of vertices \mathcal{V} , defined according to the Triggering Model. For each i , let \mathcal{G}'_i be the graph obtained from \mathcal{G}_i by reverting the arcs \mathcal{E}_i . Then, the penalty reduction $R_i(\mathcal{A})$ by the set of nodes \mathcal{A} using the population affected score (PA) corresponds exactly to the number of nodes influenced by set \mathcal{A} under the Triggering Model. Hence, also the expected penalty reduction $R(\mathcal{A}) = \sum_i P(i)R_i(\mathcal{A})$ is exactly equal to the influence function $\sigma(\mathcal{A})$ of [Kempe et al., 2003]. \square

Theorem 8.7.1 shows that spreading influence under the general Triggering Model can be considered a special case of our outbreak detection formalism. The problems are fundamentally related since, when spreading influence, one tries to affect as many nodes as possible, while when detecting outbreak, one wants to minimize the effect of an outbreak in the network. Secondly, note that in the example of reading blogs, it is not necessarily a good strategy to affect nodes which are very influential, as these tend to have many posts, and hence are expensive to read. In contrast to influence maximization, the notion of cost-benefit analysis is crucial to our applications.

8.7.2 Optimizing submodular functions

The fundamental result about the greedy algorithm for maximizing submodular functions in the unit-cost case goes back to [Nemhauser et al., 1978]. [Nemhauser and Wolsey, 1981] present a Mixed Integer Programming approach for maximizing submodular functions, which however does not provide running time guarantees. The first approximation results about maximizing submodular functions in the non-constant cost case were proved by [Sviridenko, 2004]. They developed an algorithm with approximation guarantee of $(1 - 1/e)$, which however requires a number of function evaluations $\Omega(B|\mathcal{V}|^4)$ in the size of the ground

set \mathcal{V} (if the lowest cost is constant). In contrast, the number of evaluations required by CELF is $\mathcal{O}(B|\mathcal{V}|)$, while still providing a constant factor approximation guarantee. A lazy greedy algorithm for optimizing submodular functions in the context of experimental design was described by [Robertazzi and Schwartz, 1989]. Their work however did not consider the case of non-constant cost functions, as we consider in this chapter.

8.7.3 Virus propagation and outbreak detection

Work on spread of diseases in networks and immunization mostly focuses on determining the value of the *epidemic threshold* [Bailey, 1975, Chakrabarti et al., 2007b, 2008], a critical value of the virus transmission probability above which the virus creates an epidemic. Several strategies for immunization have also been proposed. Uniform node immunization and targeted immunization of high degree nodes was proposed by [Pastor-Satorras and Vespignani, 2002], acquaintance immunization, which focuses on highly connected nodes by [Cohen et al., 2003], and immunization based on spectral properties of the network was proposed by [Giakkoupis et al., 2005]. In the context of our work, uniform immunization strategy corresponds to randomly placing sensors in a water network. Similarly, targeted immunization corresponds to selecting blogs based on their in- or out-degree. As we have seen in Figures 8.6 and 8.13, both strategies perform much worse than direct optimization of the *population affected* criterion.

8.7.4 Information cascades and blog networks.

Cascades have been studied for many years by sociologists concerned with the *diffusion of innovation* [Rogers, 1995]; more recently, cascades were used for studying viral marketing [Goldenberg et al., 2001, Leskovec et al., 2006a], selecting trendsetters in social networks [Richardson and Domingos, 2002b], and explaining trends in blogspace [Gruhl et al., 2004, Kumar et al., 2003]. Studies of blogspace either spend effort mining topics from posts [Gruhl et al., 2004] or consider only the properties of blogspace as a graph of unlabeled URLs [Kumar et al., 2003]. Recently, [Leskovec et al., 2007d] studied the properties and models of information cascades in blogs. While previous work either focused on empirical analyses of information propagation and/or provided models for it, we develop a general methodology for node selection in networks while optimizing a given criterion.

8.7.5 Water distribution network monitoring.

A large number of approaches have been proposed for optimizing water sensor networks (*c.f.*, [Berry et al., 2006, Guan et al., 2006, Ostfeld and Salomons, 2004, Dorini et al., 2006] for a concise overview of the prior literature). Most of these approaches are only applicable to small networks up to approximately 500 nodes. Many approaches are based on heuristics (such as genetic algorithms [Ostfeld and Salomons, 2004], cross-entropy selection [Dorini et al., 2006], predator-prey heuristics [Gueli, 2006], etc.) that cannot provide provable performance guarantees about the solutions. Closest to ours is an approach by [Berry et al., 2006], who equate the placement problem with a p -median problem, and make use of a large toolset of existing algorithms for this problem. The problem instances solved by [Berry et al., 2006] are a factor 72 smaller than the instances considered in this chapter. In order to obtain bounds for the quality of the generated placements, the approach in [Berry et al., 2006] needs to solve a complex (NP-hard) mixed-integer program. Our approach is the first algorithm for the water network placement problem, which is

guaranteed to provide solutions which achieve at least a constant fraction of the optimal solution within polynomial time. Additionally, it handles orders of magnitude larger problem instances than previously considered.

8.8 Conclusions

In this chapter, we presented a novel methodology for selecting nodes to detect outbreaks of dynamic processes spreading over a graph. We showed that many important objective functions, such as detection time, likelihood and affected population are *submodular*. We then developed the CELF algorithm, which exploits submodularity to find *near-optimal* node selections – the obtained solutions are guaranteed to achieve at least a fraction of $\frac{1}{2}(1 - 1/e)$ of the optimal solution, even in the more complex case where every node can have an *arbitrary* cost. Our CELF algorithm is up to 700 times faster than standard greedy algorithm. We also developed novel online bounds on the quality of the solution obtained by *any* algorithm. We used these bounds to prove that the solutions we obtained in our experiments achieve 90% of the optimal score (which is intractable to compute).

We extensively evaluated our methodology on two large real-world problems: (a) detection of contaminations in the largest *city water distribution* network considered so far in the literature, and (b) selection of *informative blogs* in a network of more than 10 million posts. We showed how our CELF algorithm greatly outperforms intuitive heuristics. We also demonstrated that our methodology can be used to study complex application-specific questions such as multicriteria tradeoff, cost-sensitivity analyses and generalization behavior. In addition to demonstrating the effectiveness of our method, we obtained some counterintuitive results about the problem domains, such as the fact that the popular blogs might not be the most effective way to catch relevant information cascades.

We are convinced that the methodology introduced in this chapter can apply to many other applications, such as computer network security, immunization and viral marketing.

Part 2 – Network cascades: Conclusion

In this chapter we presented our work on the processes that spread from node to node in the network like viruses. We investigated two such examples where propagations naturally form cascades and we were able to directly measure and observe them on a large scale.

Observations: We found that most cascades are small, but large bursts can occur; that cascade sizes follow a heavy-tailed distribution; that the frequency of different cascade subgraphs depends on the product or blog type; and that these frequencies do *not simply decrease monotonically* for denser subgraphs, but rather reflect more subtle features of the domain in which the diffusion and propagations are operating.

Models: Moreover, we were able to obtain a number of interesting insights into how viral marketing and information propagation on the blogosphere work that challenge common assumptions made in epidemic and rumor propagation modeling. For example, on a large dataset we showed the *diminishing returns* property of human adoption curve as opposed to critical threshold that is often assumed. Moreover, it is frequently assumed in epidemic models that individuals have equal probability of being infected every time they interact. Contrary to this we observe that the probability of infection *decreases* with repeated interaction. Marketers should take heed that providing excessive incentives for customers to recommend products could backfire by weakening the credibility of the very same links they are trying to take advantage of.

On the information propagation side of things we also analyzed one of the largest available collections of blog information, trying to find how blogs behave and how information propagates through the blogosphere. In contrast with viral marketing stars and chains are basic components of blog cascades, with stars being more common.

Algorithms: We presented a novel methodology for selecting nodes to detect outbreaks of dynamic processes spreading over a graph. We showed that many important objective functions, such as detection time, likelihood and affected population are *submodular*. We then developed the CELF algorithm, which exploits submodularity to find near-optimal node selections. Our CELF algorithm is up to *700 times faster* than a standard greedy algorithm. We also developed novel online bounds on the quality of the solution obtained by *any* algorithm. We used these bounds to prove that the solutions we obtained in our experiments achieve 90% of the optimal score (which is intractable to compute). We extensively evaluated our methodology on two large real-world problems: (a) detection of contaminations in the *largest water distribution* network considered so far, and (b) selection of informative blogs in a network of more than 10 million posts. We showed that the proposed CELF algorithm greatly outperforms intuitive heuristics. We also demonstrated that our methodology can be used to study complex application-specific questions such as multicriteria tradeoff, cost-sensitivity analyses and generalization behavior.

Part III

Large data

What are the properties of world's social network?

How to quantify network community structure?

**How to predict web search result quality without
page content?**

Part 3 – Large data: Overview

In the last part of the thesis we present case studies that demonstrate the value and importance of working with large data. We examine three different aspects of analysis of very large networks: (a) observations on the largest social network ever analyzed, (b) modeling of network community structure, and (c) machine learning for web search. In all three cases there will be two common topics. First, we show how large amounts of data give us opportunities to observe phenomena that are practically invisible when working with small data, and how this leads to new counterintuitive discoveries. Second, we will be exploring how microscopic behaviors can be used to make statements about the global structure.

Observations: We present a study of a month of high-level communication activities within the whole of the Microsoft Messenger instant-messaging network. We examine patterns that emerge from the collective dynamics of large numbers of people. The dataset contains 255 billion messages in 30 billion conversations among 240 million people. From the data, we construct a communication graph with 180 million nodes and 1.3 billion undirected edges, creating the *largest social network* analyzed to date. We report on multiple aspects of the dataset and synthesized graph. We investigate on a planetary-scale the oft-cited report that people are separated by “six degrees of separation” and find that the average path length among Messenger users is 6.6. We also find that the graph is well-connected and robust to node removal.

Models: Second example that “large data matters” is our work on statistical properties of community structure in networks. Researchers commonly assume the presence of “network communities”, where the intuition is that networks contain clusters of nodes that interact more strongly with each other than with the remainder of the network. Most often this has been only verified on very small networks of hundreds of nodes. On the other hand, we look at networks of millions of nodes, and find very different network structure. We find that network communities exist only up to a *size scale of ≈ 100 nodes*, and beyond that point small communities merge into the large densely interlinked network core of very little community structure. This closely agrees with Dunbar’s observation [Dunbar, 1998] that predicted 150 is the upper bound on the human community size. We develop and analyze models and explanations why such structures occur in real life.

Algorithms: We study how linking relationships among web pages can be leveraged as sources of information in methods for ranking search results. We show how local structure of the web graph can be used to make globally accurate predictions about relevancy of web pages. We introduce *web projections*, where we extract context sensitive subgraphs of the web, and then use machine learning to construct predictive models that consider graphical properties as evidence. We describe the method and present experiments that illustrate the construction of predictive models of *search result quality* and *user modeling*.

Chapter 9

MSN Messenger communication network

We present a study of anonymized data capturing a month of high-level communication activities within the whole of the Microsoft Messenger instant-messaging system. We examine characteristics and patterns that emerge from the collective dynamics of large numbers of people, rather than the actions and characteristics of individuals. The dataset contains summary properties of 30 billion conversations among 240 million people. From the data, we construct a communication graph with 180 million nodes and 1.3 billion undirected edges, creating the largest social network constructed and analyzed to date. We report on multiple aspects of the dataset and synthesized graph. We find that the graph is well-connected and robust to node removal. We investigate on a planetary-scale the oft-cited report that people are separated by “six degrees of separation” and find that the average path length among Messenger users is 6.6. We also find that people tend to communicate more with each other when they have similar age, language, and location, and that cross-gender conversations are both more frequent and of longer duration than conversations with the same gender.

9.1 Introduction

Large-scale web services provide unprecedented opportunities to capture and analyze behavioral data on a planetary scale. We discuss findings drawn from aggregations of anonymized data representing one month (June 2006) of high-level communication activities of people using the Microsoft Messenger instant-messaging (IM) network. We did not have nor seek access to the content of messages. Rather, we consider structural properties of a communication graph and study how structure and communication relate to user demographic attributes, such as gender, age, and location. The data set provides a unique lens for studying patterns of human behavior on a wide scale.

We explore a dataset of 30 billion conversations generated by 240 million distinct users over one month. We found that approximately 90 million distinct Messenger accounts were accessed each day and that these users produced about 1 billion conversations, with approximately 7 billion exchanged messages per day. 180 million of the 240 million active accounts had at least one conversation on the observation period. We found that 99% of the conversations occurred between 2 people, and the rest with greater numbers of participants. To our knowledge, our investigation represents the largest and most comprehensive study

to date of presence and communications in an IM system. A recent report [IDC Market Analysis, 2005] estimated that approximately 12 billion instant messages are sent each day. Given the estimate and the growth of IM, we estimate that we captured approximately half of the world's IM communication during the observation period.

We created an undirected *communication network* from the data where each user is represented by a node and an edge is placed between users if they exchanged at least one message during the month of observation. The network represents accounts that were active during June 2006. In summary, the communication graph has 180 million nodes, representing users who participated in at least one conversation, and 1.3 billion undirected edges among active users, where an edge indicates that a pair of people communicated. We note that this graph should be distinguished from a buddy graph where two people are connected if they appear on each other's contact lists. The buddy graph for the data contains 240 million nodes and 9.1 billion edges. On average each account has approximately 50 buddies on a contact list.

To highlight several of our key findings, we discovered that the communication network is well connected, with 99.9% of the nodes belonging to the largest connected component. We evaluated the oft-cited finding by Travers and Milgram that any two people are linked to one another on average via a chain with "6-degrees-of-separation" [Milgram, 1967, Travers and Milgram, 1969]. We found that the average shortest path length in the Messenger network is 6.6 (median 6), which is half a link more than the path length measured in the classic study. However, we also found that longer paths exist in the graph, with lengths up to 29. We observed that the network is well clustered, with a clustering coefficient [Watts and Strogatz, 1998] that decays with exponent -0.37 . This decay is significantly lower than the value we had expected given prior research [Ravasz and Barabási, 2003]. We found strong *homophily* [McPherson et al., 2001, Rogers and Bhowmik, 1970] among users; people have more conversations and converse for longer durations with people who are similar to themselves. We find the strongest homophily for the language used, followed by conversants' geographic locations, and then age. We found that homophily does not hold for gender; people tend to converse more frequently and with longer durations with the opposite gender. We also examined the relation between communication and distance, and found that the number of conversations tends to decrease with increasing geographical distance between conversants. However, communication links spanning longer distances tend to carry more and longer conversations.

9.2 Instant Messaging

The use of IM has become widely adopted in personal and business communications. IM clients allow users fast, near-synchronous communication, placing it between synchronous communication mediums, such as real-time voice interactions like telephone, and asynchronous communication mediums like mail or email [Voida et al., 2002]. IM users exchange short text messages with one or more users from their list of contacts, who have to be on-line and logged into the IM system at the time of interaction. As conversations and messages exchanged within them are usually very short, it has been observed that users employ informal language, loose grammar, numerous abbreviations, with minimal punctuation [Nardi et al., 2000]. Contact lists are commonly referred to as *buddy lists* and users on the lists are referred to as *buddies*.

SYMBOL	DESCRIPTION
d_i	Duration of i^{th} conversation
m_i	Number of exchanged messages in i^{th} conversation
l_i	Geographical distance between the a pair of users in i^{th} conversation
$m_{u,i}$	Number of exchanged messages in i^{th} conversation of user u
$C_{a,b}$	Set of all conversations between users of age a and b
$C_{g,h}$	Set of all conversations between users of genders g and h
ti_j	Time of j^{th} login of a user
to_j	Time of j^{th} logout of a user
$ts_{u,i}$	Start time of i^{th} conversation of user u
$te_{u,i}$	End time of i^{th} conversation of user u

Table 9.1: Table of symbols.

9.2.1 Research on Instant Messaging

Several studies on smaller datasets are related to this work. Avrahami and Hudson [Avrahami and Hudson, 2006] explored communication characteristics of 16 IM users. Similarly, Shi et al. [Shi et al., 2007] analyzed IM contact lists submitted by users to a public website and explored a static contact network of 140,000 people. Recently, Xiao et al. [Xiao et al., 2007] investigated IM traffic characteristics within a large organization with 400 users of Messenger. Our study differs from the latter study in that we analyze the *full* Messenger population over a one month period, capturing the interaction of user demographic attributes, communication patterns, and network structure.

9.2.2 Data description

To construct the Microsoft Instant Messenger communication dataset, we combined three different sources of data: (1) user demographic information, (2) time and user stamped events describing the presence of a particular user, and (3) communication session logs, where, for all participants, the number of exchanged messages and the periods of time spent participating in sessions is recorded.

We use the terms *session* and *conversation* interchangeably to refer to an IM interaction among two or more people. Although the Messenger system limits the number of people communicating at the same time to 20, people can enter and leave a conversation over time. We note that, for large sessions, people can come and go over time, so conversations can be long with many different people participating. We observed some very long sessions with more than 50 participants joining over time.

All of our data was anonymized; we had no access to personally identifiable information. Also, we had no access to text of the messages exchanged or any other information that could be used to uniquely identify users. We focused on analyzing high-level characteristics and patterns that emerge from the collective dynamics of 240 million people, rather than the actions and characteristics of individuals. The analyzed data can be split into three parts: *presence data*, *communication data*, and *user demographic information*:

- **Presence events:** These include login, logout, first ever login, add, remove and block a buddy, add unregistered buddy (invite new user), change of status (busy, away, be-right-back, idle, etc.). Events are user and time stamped.

- **Communication:** For each user participating in the session, the log contains the following tuple: session id, user id, time joined the session, time left the session, number of messages sent, number of messages received.
- **User data:** For each user, the following self-reported information is stored: age, gender, location (country, ZIP), language, and IP address. We use the IP address to decode the geographical coordinates, which we then use to position users on the globe and to calculate distances.

We gathered data for 30 days of June 2006. Each day yielded about 150 gigabytes of compressed text logs (4.5 terabytes in total). Copying the data to a dedicated eight-processor server with 32 gigabytes of memory took 12 hours. Our log-parsing system employed a pipeline of four threads that parse the data in parallel, collapse the session join/leave events into sets of conversations, and save the data in a compact compressed binary format. This process compressed the data down to 45 gigabytes per day. Processing the data took an additional 4 to 5 hours per day.

A special challenge was to account for missing and dropped events, and session “id recycling” across different IM servers in a server farm. As part of this process, we closed a session 48 hours after the last leave session event. We closed sessions automatically if only one user was left in the conversation.

9.3 Usage & population statistics

We shall first review several statistics drawn from aggregations of users and their communication activities.

9.3.1 Levels of activity

Over the observation period, 242,720,596 users logged into Messenger and 179,792,538 of these users were actively engaged in conversations by sending or receiving at least one IM message. Over the month of observation, 17,510,905 new accounts were activated. As a representative day, on June 1 2006, there were almost 1 billion (982,005,323) different sessions (conversations among any number of people), with more than 7 billion IM messages sent. Approximately 93 million users logged in with 64 million different users becoming engaged in conversations on that day. Approximately 1.5 million new users that were not registered within Microsoft Messenger were invited to join on that particular day.

We consider event distributions on a per-user basis in Figure 9.1. The number of logins per user, displayed in Figure 9.1(a), follows a heavy-tailed distribution with exponent 3.6. We note spikes in logins at 20 minute and 15 second intervals, which correspond to an auto-login function of the IM client. As shown in Figure 9.1(b), many users fill up their contact lists rather quickly. The spike at 600 buddies undoubtedly reflects the maximal allowed length of contact lists.

Figure 9.2(a) displays the number of users per session. In Messenger, multiple people can participate in conversations. We observe a peak at 20 users, the limit on the number of people who can participate simultaneously in a conversation. Figure 9.2(b) shows the distribution over the session durations, which can be modeled by a power law distribution with exponent 3.6.

Next, we examine the distribution of the durations of periods of time when people are logged on to the system. Let (t_{ij}, t_{oj}) denote a time ordered $(t_{ij} < t_{oj} < t_{ij+1})$ sequence of online and offline times

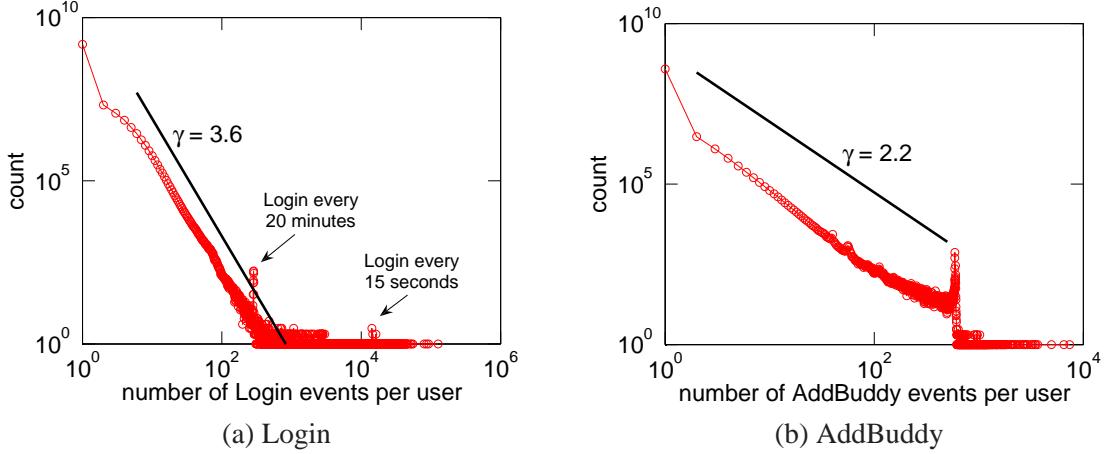


Figure 9.1: Distribution of the number of events per user. (a) Number of logins per user. (b) Number of buddies added per user.

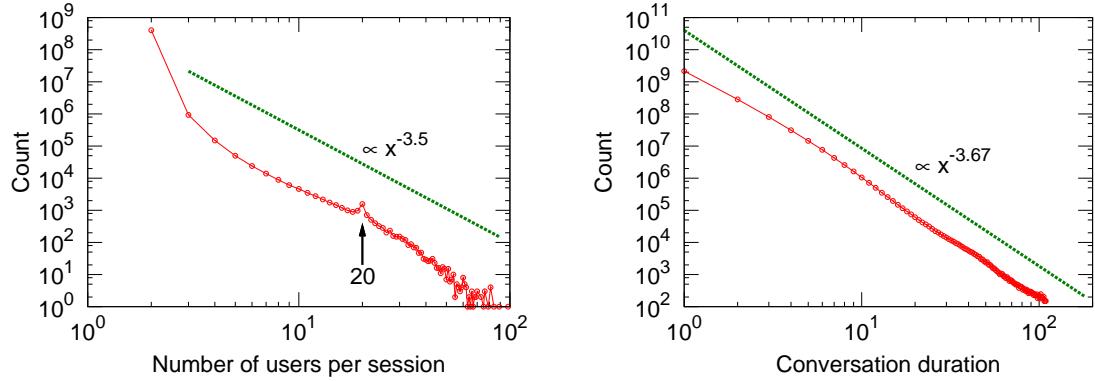


Figure 9.2: (a) Distribution of the number of people participating in a conversation. (b) Distribution of the durations of conversations. The spread of durations can be described by a power law distribution.

of a user, where t_{ij} is the time of the j th login, and to_j is the corresponding logout time. Figure 9.3(a) plots the distribution of $to_j - t_{ij}$ over all j over all users. Similarly, Figure 9.3(b) shows the distribution of the periods of time when users are logged off, *i.e.*, $t_{ij+1} - to_j$ over all j and over all users. Fitting the data to power law distributions reveals exponents of 1.77 and 1.3, respectively. The data shows that durations of being online tend to be shorter and decay faster than durations that users are offline. We also notice periodic effects of login durations of 12, 24, and 48 hours, reflecting daily periodicities. We observe similar periodicities for logout durations at multiples of 24 hours.

Weekly dynamics of MSN Messenger is also quite interesting. Figure 9.4 shows the number of logins, status change and add buddy events by day of the week over a period of 5 weeks starting in June 2006. We count the number of particular events per day of the week, and we use the data from 5 weeks to compute the error bars. Figure 9.4(a) shows the average number of logins per day of the week over a 5 week period. Note that number of login events is larger than the number of distinct users logging in, since a user can login multiple times a day. Figure 9.4(b) plots the average number of status change events per day of the

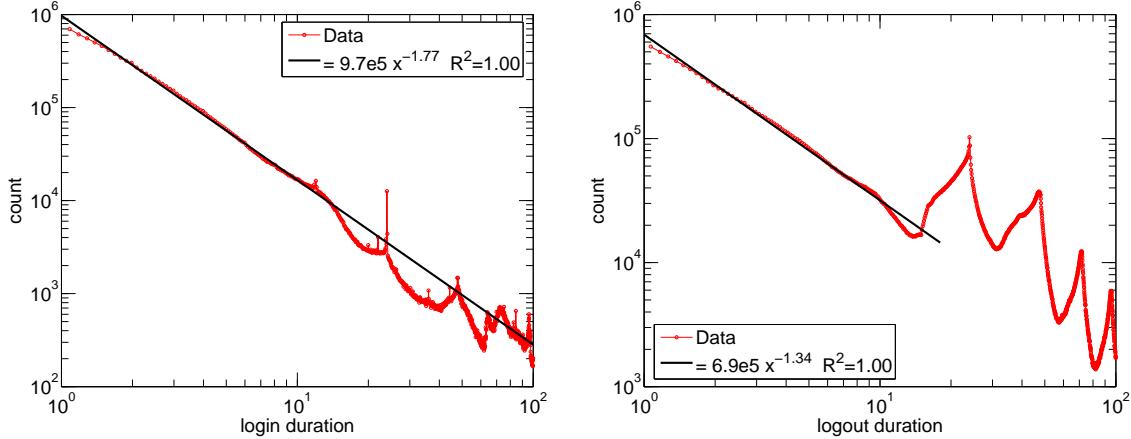


Figure 9.3: (a) Distribution of login duration. (b) Duration of times when people are not logged into the system (times between logout and login).

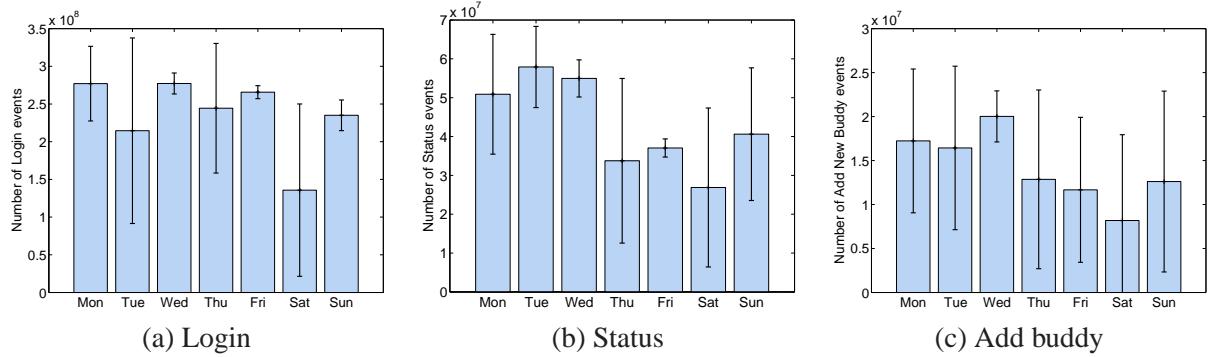


Figure 9.4: Number of events per day of the week. We collected the data over a period of 5 weeks starting on May 29 2006.

week. Status events include a group of 8 events describing the current status of the users, *i.e.*, away, be right back, online, busy, idle, at lunch, and on the phone. Last, Figure 9.4(c) shows the average number of add buddy events per day of the week. Add buddy event is triggered every time user adds a new contact to their contact list.

9.3.2 Demographic characteristics of the users

We compared the demographic characteristics of the Messenger population with 2005 world census data and found differences between the statistics for age and gender. The visualization of this comparison displayed in Figure 9.5 shows that users with reported ages in the 15–35 span of years are strongly overrepresented in the active Messenger population. Focusing on the differences by gender, females are overrepresented for the 10–14 age interval. For male users, we see overall matches with the world population for age spans 10–14 and 35–39; for women users, we see a match for ages in the span of 30–34. We note that 6.5% of the population did not submit an age when creating their Messenger accounts.

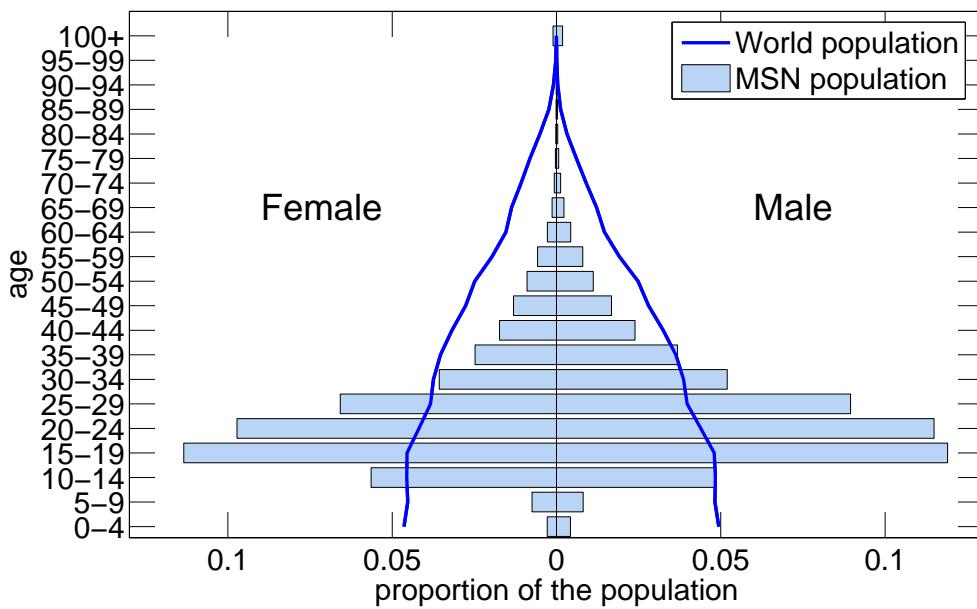


Figure 9.5: World and Messenger user population age pyramid. Ages 15–30 are overrepresented in the Messenger population.

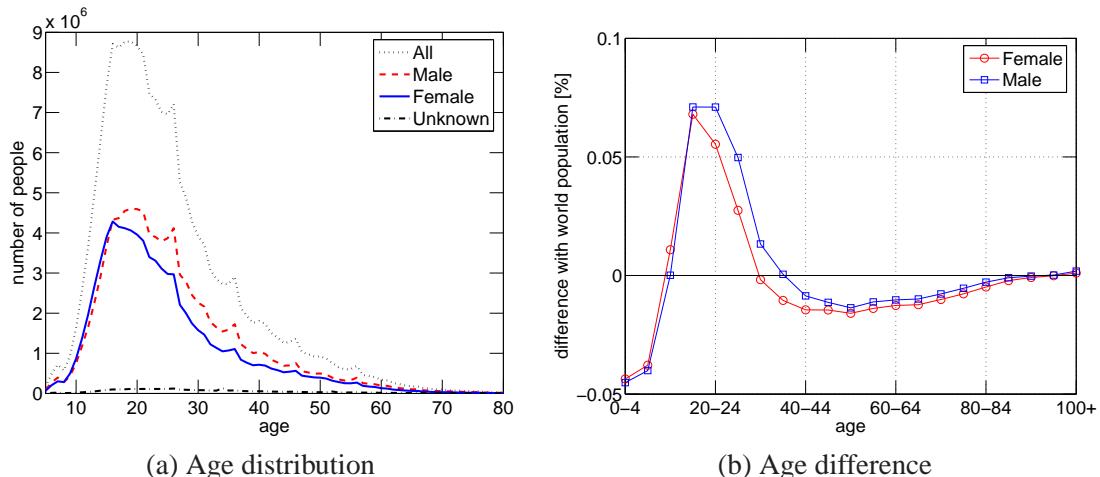


Figure 9.6: Distribution of self-reported ages of Messenger users and the difference of ages of Messenger population with the world population. (a) Age distribution for all users, females, males and unknown users. (b) Relative difference of Messenger population and the world population. Ages 15–30 are over-represented in the Messenger user population.

To further illustrate the points above Figure 9.6 shows self-reported user age distribution and the percent difference of particular age-group between MSN and the world population. The distribution is skewed to the right and has a mode at age of 18. We also note that the distribution has exponential tails.

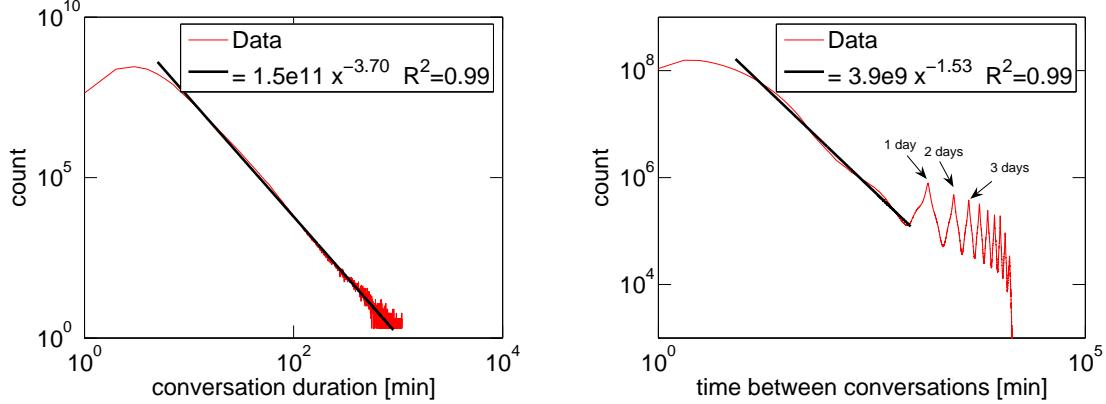


Figure 9.7: Temporal characteristics of conversations. (a) Average conversation duration per user; (b) time between conversations of users.

9.4 Communication characteristics

We now focus on characteristics and patterns with communications. We limit the analysis to conversations between two participants, which account for 99% of all conversations.

We first examine the distributions over conversation durations and times between conversations. Let user u have C conversations in the observation period. Then, for every conversation i of user u we create a tuple $(ts_{u,i}, te_{u,i}, m_{u,i})$, where $ts_{u,i}$ denotes the start time of the conversation, $te_{u,i}$ is the end time of the conversation, and $m_{u,i}$ is the number of exchanged messages between the two users. We order the conversations by their start time ($ts_{u,i} < ts_{u,i+1}$). Then, for every user u , we calculate the average conversation duration $\bar{d}(u) = \frac{1}{C} \sum_i te_{u,i} - ts_{u,i}$, where the sum goes over all the u 's conversations. Figure 9.7(a) shows the distribution of $\bar{d}(u)$ over all the users u . We find that the conversation length can be described by a heavy-tailed distribution with exponent -3.7 and a mode of 4 minutes.

Figure 9.7(b) shows the intervals between consecutive conversations of a user. We plot the distribution of $ts_{u,i+1} - ts_{u,i}$, where $ts_{u,i+1}$ and $ts_{u,i}$ denote start times of two consecutive conversations of user u . The power law exponent of the distribution over intervals is -1.5 . This result is similar to the temporal distribution for other kinds of human communication activities, e.g., waiting times of emails and letters before a reply is generated [Barabási, 2005]. The exponent can be explained by a priority-queue model where tasks of different priorities arrive and wait until all tasks with higher priority are addressed. This model generates a task waiting time distribution described by a power law with exponent -1.5 .

However, the total number of conversations between a pair of users (Figure 9.8(a)), and the total number of exchanged messages between a pair of users (Figure 9.8(b)) does not seem to follow a power law. The distribution seems still to be heavy tailed but not power law. The fits represent the MLE estimates of a log-normal distribution.

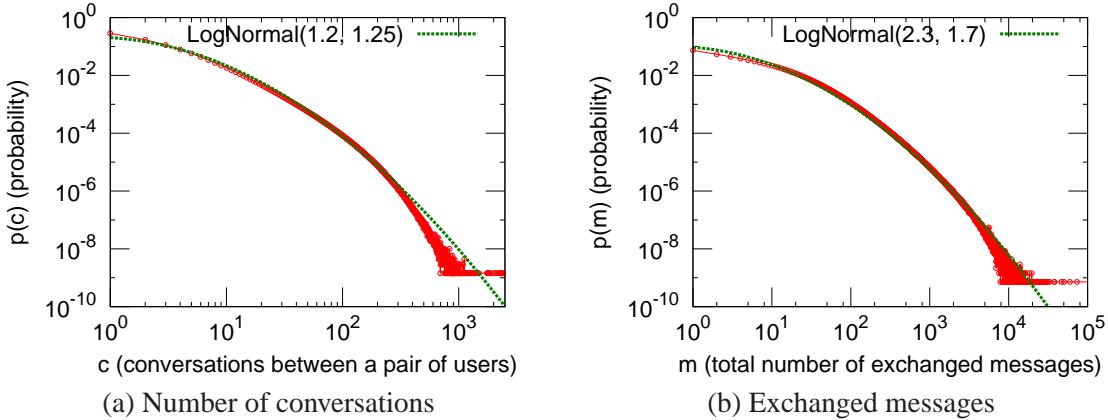


Figure 9.8: Conversation statistics: (a) Number of conversations of a user in a month; (b) Number of messages exchanged per conversation;

9.5 Communication demographics

Next we examine the interplay of communication and user demographic attributes, *i.e.*, how geography, location, age, and gender influence observed communication patterns.

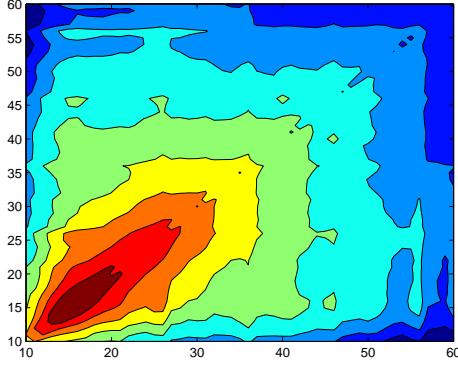
9.5.1 Communication by age

We sought to understand how communication among people changes with the reported ages of participating users. Figures 9.9(a)-(d) use a heat-map visualization to communicate properties for different age–age pairs. The rows and columns represent the ages of both parties participating, and the color at each age–age cell captures the logarithm of the value for the pairing. The color spectrum extends from blue (low value) through green, yellow, and onto red (the highest value). Because of potential misreporting at very low and high ages, we concentrate on users with self-reported ages that fall between 10 and 60 years.

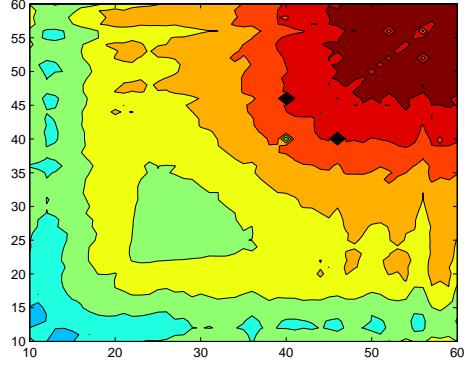
Let a tuple (a_i, b_i, d_i, m_i) denote the i th conversation in the entire dataset that occurred among users of ages a_i and b_i . The conversation had a duration of d_i seconds during which m_i messages were exchanged. Let $C_{a,b} = \{(a_i, b_i, d_i, m_i) : a_i = a \wedge b_i = b\}$ denote a set of all conversations between users of ages a and b , respectively.

Figure 9.9(a) shows the number of conversations among people of different ages. For every pair of ages (a, b) the color indicates the size of set $C_{a,b}$, *i.e.*, the number of different conversations between users of ages a and b . We note that, as the notion of a conversation is symmetric, the plots are symmetric. Most conversations occur between people of ages 10 to 20. The diagonal trend indicates that people tend to talk to people of similar age. This is true especially for age groups between 10 and 30 years. We shall explore this observation in more detail in Section 9.6.

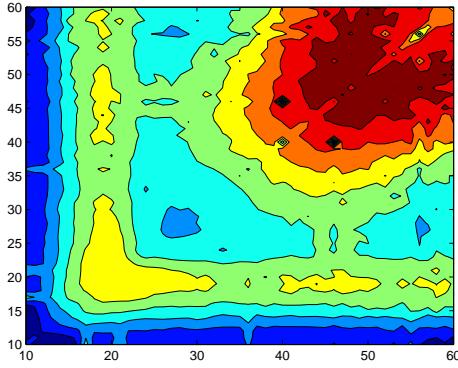
Figure 9.9(b) displays a heat map for the average conversation duration, computed as $\frac{1}{|C_{a,b}|} \sum_{i \in C_{a,b}} d_i$. We note that older people tend to have longer conversations. We observe a similar phenomenon when plotting the average number of exchanged messages per conversation, computed as $\frac{1}{|C_{a,b}|} \sum_{i \in C_{a,b}} m_i$, displayed in Figure 9.9(c). Again, we find that older people exchange more messages, and we observe



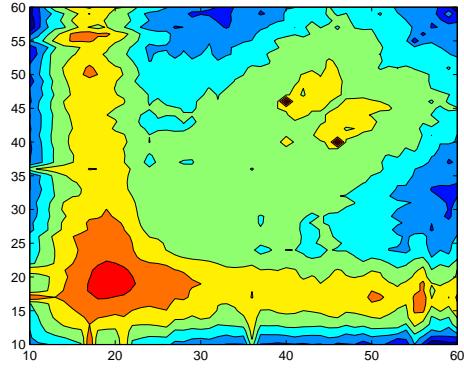
(a) Number of conversations



(b) Conversation duration



(c) Messages per conversation



(d) Messages per unit time

Figure 9.9: Communication characteristics of users by reported age. We plot age vs. age and the color (z-axis) represents the intensity of communication.

a dip for ages 25–45 and a slight peak for ages 15–25. Figure 9.9(d) displays the number of exchanged messages per unit time; for each age pair, (a, b) , we measure $\frac{1}{|C_{a,b}|} \sum_{i \in C_{a,b}} \frac{m_i}{d_i}$. Here, we see that younger people have faster-paced dialogs, while older people exchange messages at a slower pace.

We note that the younger population (ages 10–35) are strongly biased towards communicating with people of a similar age (diagonal trend in Figure 9.9(a)), and that users who report being of ages 35 years and above tend to communicate more evenly across ages (rectangular pattern in Fig. 9.9(a)). Moreover, older people have conversations of the longest durations, with a “valley” in the duration of conversations for users of ages 25–35. Such a dip may represent shorter, faster-paced and more intensive conversations associated with work-related communications, versus more extended, slower, and longer interactions associated with social discourse.

9.5.2 Communication by gender

We report on analyses of properties of pairwise communications as a function of the self-reported gender of users in conversations in Table 9.2. Let $C_{g,h} = \{(g_i, h_i, d_i, m_i) : g_i = g \wedge h_i = h\}$ denote a set

	Unknown	Female	Male
Unknown	1.3	3.6	3.7
Female		21.3	49.9
Male			20.2

(a) Conversations

	Unknown	Female	Male
Unknown	277	301	277
Female		275	304
Male			252

(b) Conversation duration

	Unknown	Female	Male
Unknown	5.7	7.1	6.7
Female		6.6	7.6
Male			5.9

(c) Exchanged messages per conversation

	Unknown	Female	Male
Unknown	1.25	1.42	1.38
Female		1.43	1.50
Male			1.42

(d) Conversation intensity

Table 9.2: Cross-gender communication. Data is based on all two-person conversations from June 2006.

(a) Percentage of conversations among users of different self-reported gender; (b) average conversation length in seconds; (c) number of exchanged messages per conversation; (d) number of exchanged messages per minute of conversation.

of conversations where the two participating users are of genders g and h . Note that g takes 3 possible values: female, male, and unknown (unreported).

Table 9.2(a) relays $|C_{g,h}|$ for combinations of genders g and h . The table shows that approximately 50% of conversations occur between male and female and 40% of the conversations occur among users of the same gender (20% for each). A small number of conversations occur between people who did not reveal their gender.

Similarly, Table 9.2(b) shows the average conversation length in seconds, broken down by the gender of conversant, computed as $\frac{1}{|C_{g,h}|} \sum_{i \in C_{g,h}} d_i$. We find that male–male conversations tend to be shortest, lasting approximately 4 minutes. Female–female conversations last 4.5 minutes on the average. Female–male conversations have the longest durations, taking more than 5 minutes on average. Beyond taking place over longer periods of time, more messages are exchanged in female–male conversations. Table 9.2(c) lists values for $\frac{1}{|C_{g,h}|} \sum_{i \in C_{g,h}} m_i$ and shows that, in female–male conversations, 7.6 messages are exchanged per conversation on the average as opposed to 6.6 and 5.9 for female–female and male–male, respectively. Table 9.2(d) shows the communication intensity computed as $\frac{1}{|C_{g,h}|} \sum_{i \in C_{g,h}} \frac{m_i}{d_i}$. The number of messages exchanged per minute of conversation for male–female conversations is higher at 1.5 messages per minute than for cross-gender conversations, where the rate is 1.43 messages per minute.

We examined the number of *communication ties*, where a tie is established between two people when they exchange at least one message during the observation period. We computed 300 million male–male ties, 255 million female–female ties, and 640 million cross-gender ties. The Messenger population consists of 100 million males and 80 million females by self report. These findings demonstrate that ties are not heavily gender biased; based on the population, random chance predicts 31% male–male, 20% female–female, and 49% female–male links. We observe 25% male–male, 21% female–female, and 54% cross-gender links, thus demonstrating a minor bias of female–male links.

The results reported in Table 9.2 run counter to prior studies reporting that communication among individuals who resemble one other (same gender) occurs more often (see [McPherson et al., 2001] and

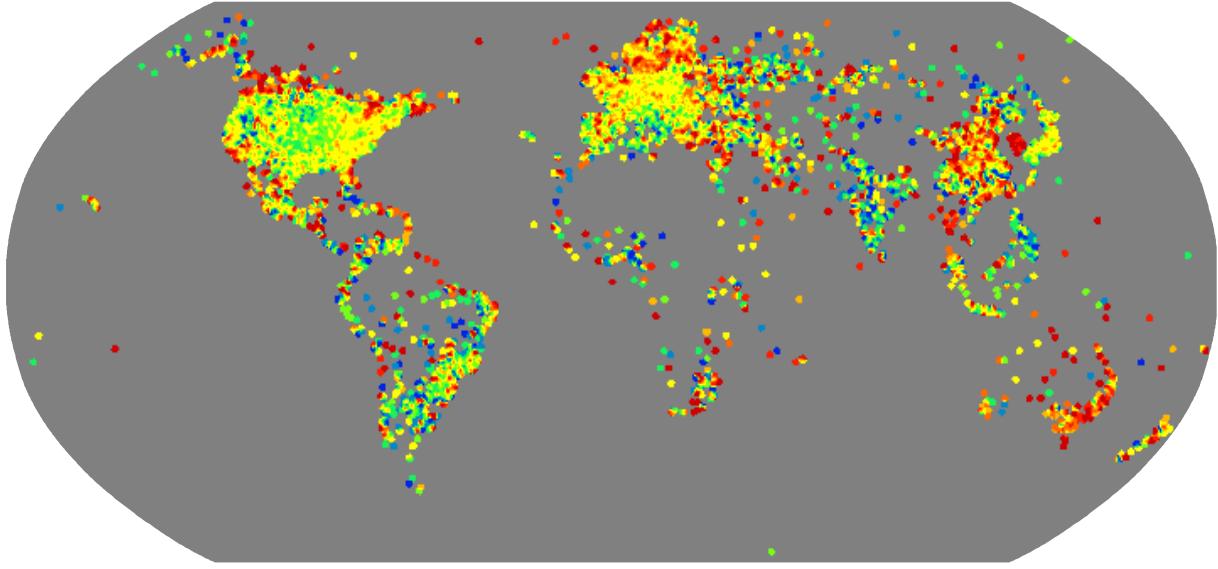


Figure 9.10: Number of users at a particular geographic location. Color represents the number of users.
Notice the map of the world appears.

references therein). We identified significant heterophily, where people tend to communicate more with people of the opposite gender. However, we note that link heterogeneity was very close to the population value [Marsden, 1987], *i.e.*, the number of same- and cross-gender ties roughly corresponds to random chance. This shows there is no significant bias in linking for gender. However, we observe that cross-gender conversations tend to be longer and to include more messages, suggesting that more effort is devoted to conversations with the opposite sex.

9.5.3 World geography and communication

We now focus on the influence of geography and distance among participants on communications. Figure 9.10 shows the geographical locations of Messenger users. The general location of the user was obtained via reverse IP lookup. We plot all latitude/longitude positions linked to the position of servers where users log into the service. The color of each dot corresponds to the logarithm of the number of logins from the respective location, again using a spectrum of colors ranging from blue (low) through green and yellow to red (high). Although the maps are built solely by plotting these positions, a recognizable world map is generated. We find that North America, Europe, and Japan are very dense, with many users from those regions using Messenger. For the rest of the world, the population of Messenger users appears to reside largely in coastal regions.

We can condition the densities and behaviors of Messenger users on multiple geographical and socioeconomic variables and explore relationships between electronic communications and other attributes. As an example, harnessed the United Nations gridded world population data to provide estimates of the number of people living in each cell. Given this data, and the data from Figure 9.10, we calculate the number of users per capita, displayed in Figure 9.12. Now we see transformed picture where several sparsely populated regions stand out as having a high usage per capita. These regions include the center of the United States, Canada, Scandinavia, Ireland, Australia, and South Korea.

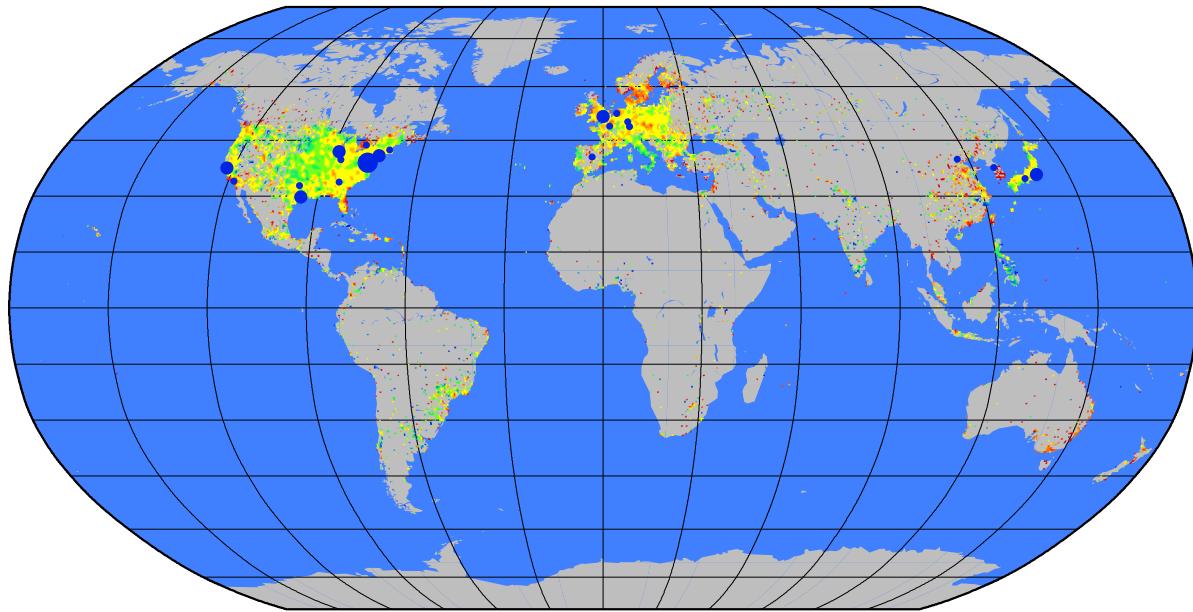


Figure 9.11: Number of users at particular geographic location superimposed on the map of the world.
Color represents the number of users.

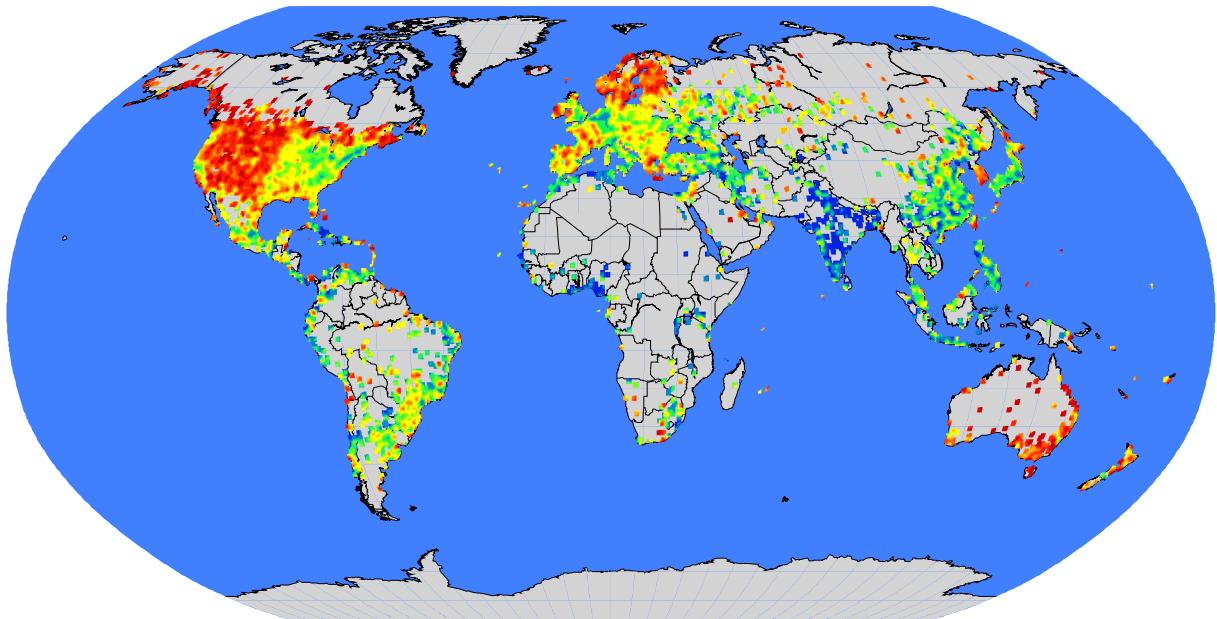


Figure 9.12: Number of Messenger users per capita. Color intensity corresponds to the number of users per capita in the cell of the grid.

Figure 9.13 shows a heat map that represents the intensities of Messenger communications on an international scale. To create this map, we place the world map on a fine grid, where each cell of the grid contains the count of the number of conversations that pass through that point by increasing the count of all cells on the straight line between the geo-locations of pairs of conversants. The color indicates the number

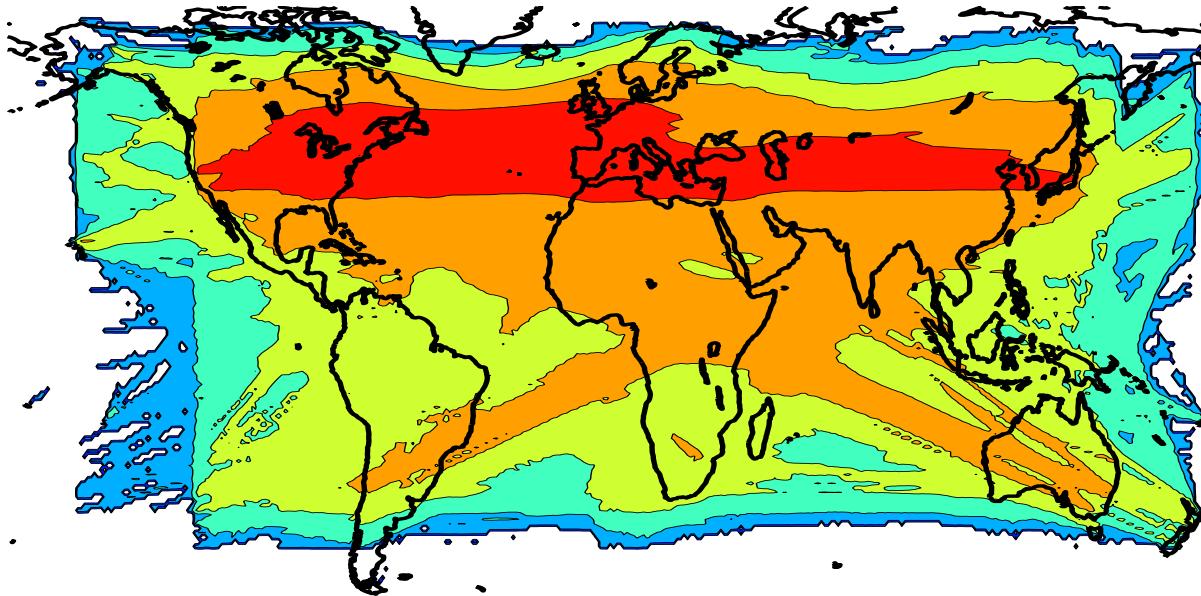


Figure 9.13: Communication heat map. For each conversation we increase the intensity of the color along the line between two conversation endpoints on the planet.

of conversations crossing each point, providing a visualization of the key flows of communication. For example, Australia and New Zealand have communications flowing towards Europe and United States. Similar flows hold for Japan. We see that Brazilian communications are weighted toward Europe and Asia. We can also explore the flows of transatlantic and US transcontinental communications.

9.5.4 Communication among countries

Communication among people within different countries also varies depending on the locations of conversants. We examine two such views. Figure 9.14 shows the top countries by the number of conversations between pairs of countries. We examined all pairs of countries with more than 10 million conversations per month. The width of edges in the figure is proportional to the logarithm of the number of conversations among the countries. We find that the United States and Spain appear to serve as hubs and that edges appear largely between historically or ethnically connected countries. As examples, Spain is connected with the Spanish speaking countries in South America, Germany links to Turkey, Portugal to Brazil, and China to Korea.

Figure 9.15 displays a similar plot where we consider country pairs by the average duration of conversations. The width of the edges are proportional to the mean length of conversations between the countries. The core of the network appears to be Arabic countries, including Saudi Arabia, Egypt, United Arab Emirates, Jordan, and Syria.

Comparing the number of active users with the country population reveals interesting findings. Table 9.3 shows the top 10 countries with the highest fraction of population using Messenger. These are mainly northern European countries and Canada. Countries with most of the users (US, Brazil) tend to have smaller fraction of population using Messenger.

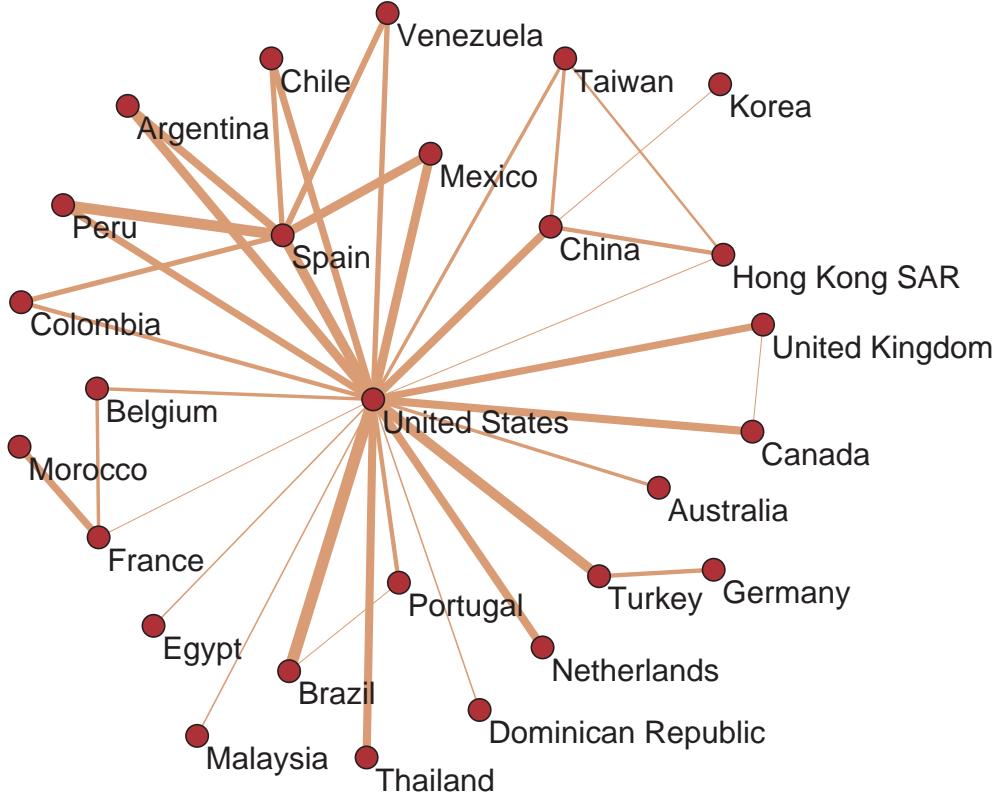


Figure 9.14: Communication among countries with at least 10 million conversations in June 2006. Edge widths correspond to logarithms of intensities of links.

Similarly, Table 9.4 shows the top 10 countries by the number of conversations per user per day. Here the countries are very diverse with Afghanistan topping the list. The Netherlands Antilles appears on top 10 list for both the fraction of the population using Messenger and the number of conversations per user.

Last, Table 9.5 shows the top 10 countries by the number of messages and minutes talking per user per day. We note that the list of the countries is similar to those in Table 9.4. Afghanistan still tops the list but now most of the talkative countries come from Eastern Europe (Serbia, Bosnia, Bulgaria, Croatia).

9.5.5 Communication and geographical distance

We were interested in how communications change as the distance between people increases. We had hypothesized that the number of conversations would decrease with geographical distance as users might be doing less coordination with one another on a daily basis, and where communication would likely require more effort to coordinate than might typically be needed for people situated more locally. We also conjectured that, once initiated, conversations among people who are farther apart would be somewhat longer as there might be a stronger need to catch up when the less-frequent conversations occurred.

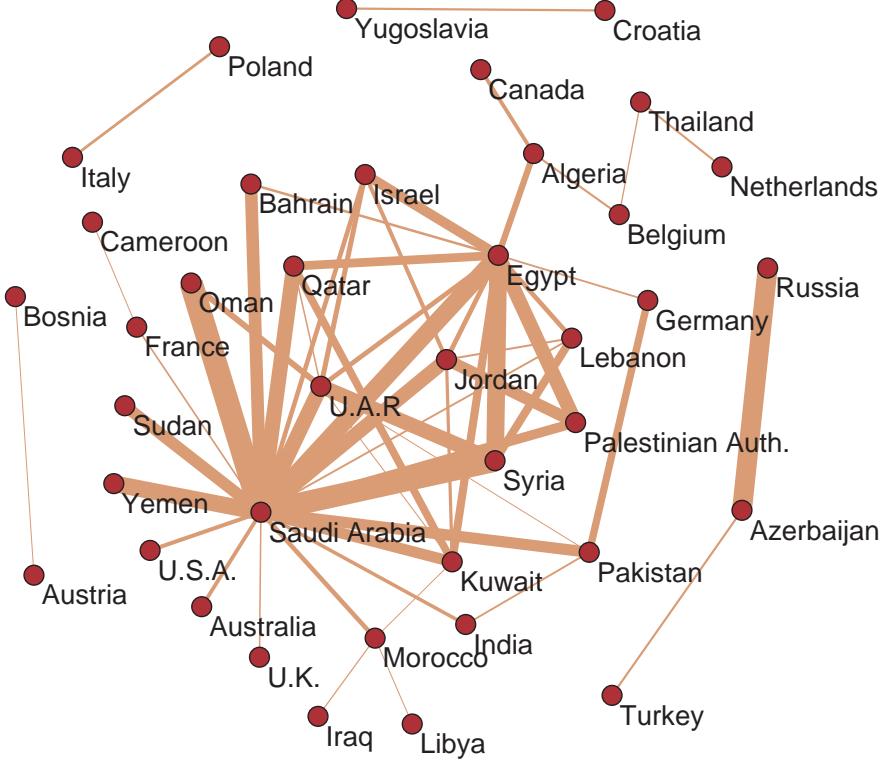


Figure 9.15: Countries by average length of the conversation. Edge widths correspond to logarithms of intensity of links.

Figure 9.16 plots the relation between communication and distance. Figure 9.16(a) shows the distribution of the number of conversations between conversants at distance l . We found that the number of conversations decreases with distance. However, we observe a peak at a distance of approximately 500 kilometers. The other peaks and drops may reveal geographical features. For example, a significant drop in communication at distance of 5,000 km (3,500 miles) may reflect the width of the Atlantic ocean or the distance between the east and west coasts of the United States. The number of links rapidly decreases with distance. This finding suggests that users may use Messenger mainly for communications with others within a local context and environment. We found that the number of exchanged messages and conversation lengths do not increase with distance (see plots (b)–(d) and (f) of Figure 9.16). Conversation duration decreases with the distance, while the number of exchanged messages remains constant before decreasing slowly. Figure 9.16(f) shows the communications per link versus the distance among participants. The plot shows that longer links, *i.e.*, connections between people who are farther apart, are more frequently used than shorter links. We interpret this finding to mean that people who are farther apart use Messenger more frequently to communicate.

In summary, we observe that the total number of links and associated conversations decreases with increasing distance among participants. The same is true for the duration of conversations, the number of exchanged messages per conversation, and the number of exchanged messages per unit time. However, the number of times a link is used tends to increase with the distance among users. This suggests that

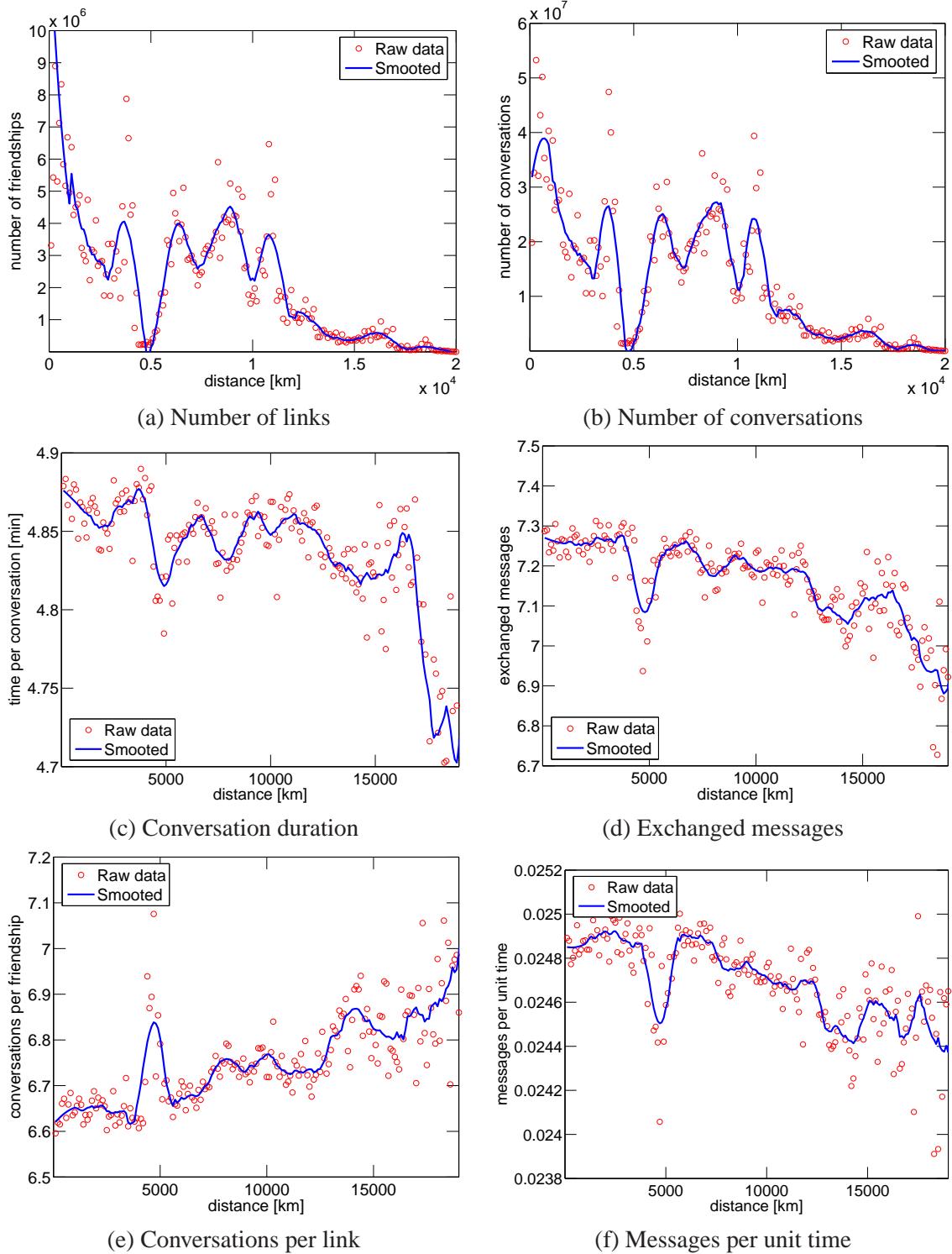


Figure 9.16: Communication with the distance. (a) Number of links (pairs of people that communicate) with the distance. (b) Number of conversations between people at particular distance. (c) Average conversation duration. (d) Number of exchanged messages per conversation. (e) Number of conversations per link (per pair of communicating users). (f) Number of exchanged messages per unit time.

Country	Fraction of population
Iceland	0.35
Spain	0.28
Netherlands	0.27
Canada	0.26
Sweden	0.25
Norway	0.25
Bahamas, The	0.24
Netherlands Antilles	0.24
Belgium	0.23
France	0.18
United Kingdom	0.17
Brazil	0.08
United States	0.08

Table 9.3: Top 10 countries with most the largest number of Messenger users. Fraction of country's population actively using Messenger.

Country	Conversations per user per day
Afghanistan	4.37
Netherlands Antilles	3.79
Jamaica	2.63
Cyprus	2.33
Hong Kong	2.27
Tunisia	2.25
Serbia	2.15
Dominican Republic	2.06
Bulgaria	2.07

Table 9.4: Top 10 countries by the number of conversations per user per day.

Country	Messages per user per day	Minutes talking per user per day
Afghanistan	32.00	20.91
Netherlands Antilles	24.12	17.43
Serbia	22.41	12.01
Bosnia and Herzegovina	22.40	11.41
Macedonia	19.52	10.46
Cyprus	19.33	12.37
Tunisia	19.17	13.54
Bulgaria	18.94	11.38
Croatia	17.78	10.05

Table 9.5: Top 10 countries by the number of messages and minutes talking per user per day.

people who are farther apart tend to converse with IM more frequently, which perhaps takes the place of more expensive long-distance voice telephony; voice might be used more frequently in lieu of IM for less expensive local communications.

Attribute	Correlation		Probability	
	Rnd	Comm	Rnd	Comm
Age	-0.0001	0.297	0.030	0.162
Gender	0.0001	-0.032	0.434	0.426
ZIP	-0.0003	0.557	0.001	0.23
County	0.0005	0.704	0.046	0.734
Language	-0.0001	0.694	0.030	0.798

Table 9.6: Correlation coefficients and probability of users sharing an attribute for random pairs of people versus for pairs of people who communicate.

9.6 Homophily of communication

We performed several experiments to measure the level at which people tend to communicate with similar people. First, we consider all 1.3 billion pairs of people who exchanged at least one message in June 2006, and calculate the similarity of various user demographic attributes. We contrast this with the similarity of pairs of users selected via uniform random sampling across 180 million users. We consider two measures of similarity: the correlation coefficient and the probability that users have the same attribute value, *e.g.*, that users come from the same countries.

Table 9.6 compares correlation coefficients of various user attributes when pairs of users are chosen uniformly at random with coefficients for pairs of users who communicate. We can see that attributes are not correlated for random pairs of people, but that they are highly correlated for users who communicate. As we noted earlier, gender and communication are slightly negatively correlated; people tend to communicate more with people of the opposite gender.

Another method for identifying association is to measure the probability that a pair of users will show an exact match in values of an attribute, *i.e.*, identifying whether two users come from the same country, speak the same language, etc. Table 9.6 shows the results for the probability of users sharing the same attribute value. We make similar observations as before. People who communicate are more likely to share common characteristics, including age, location, language, and they are less likely to be of the same gender. We note that the most common attribute of people who communicate is language. On the flip side, the amount of communication tends to decrease with increasing user dissimilarity. This relationship is highlighted in Figure 9.16, which shows how communication among pairs of people decreases with distance.

Figure 9.17 further illustrates the results displayed in Table 9.6, where we randomly sample pairs of users from the Messenger user base, and then plot the distribution over reported ages. As most of the population comes from the age group 10–30, the distribution of random pairs of people reaches the mode at those ages but there is no correlation. Figure 9.17(b) shows the distribution of ages over the pairs of people who communicate. Note the correlation, as represented by the diagonal trend on the plot, where people tend to communicate more with others of a similar age.

Next, we further explore communication patterns by the differences in the reported ages among users. Figure 9.18(a) plots the number links in the communication network vs. the age difference of the communicating pair of users. Similarly, Figure 9.18(b) plots on a log-linear scale the number of conversations in the social network with participants of varying age differences. Again we see that links and conversations are strongly correlated with the age differences among participants. Figure 9.18(c) shows the average con-

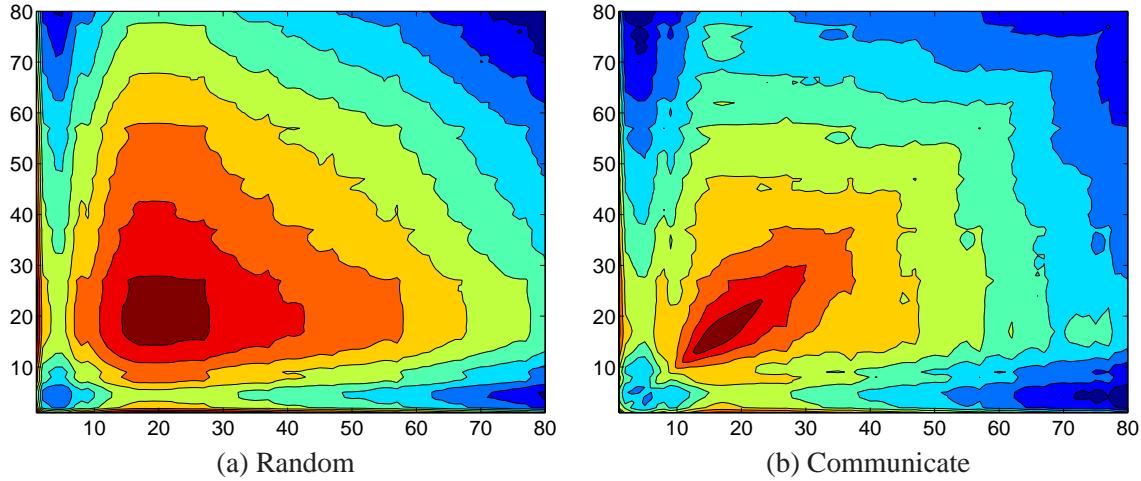


Figure 9.17: Numbers of pairs of people of different ages. (a) Randomly selected pairs of people; (b) people who communicate. Correlation between age and communication is captured by the diagonal trend.

versation duration with the age difference among the users. Interestingly, the mean conversation duration peaks at an age difference of 20 years between participants. We speculate that the peak may correspond roughly to the gap between generations.

The plots reveal that there is strong homophily in the communication network for age; people tend to communicate more with people of similar reported age. This is especially salient for the number of buddies and conversations among people of the same ages. We also observe that the links between people of similar attributes are used more often, to interact with shorter and more intense (more exchanged messages) communications. The intensity of communication decays linearly with the difference in age. In contrast to findings of previous studies, we observe that the number of cross-gender communication links follows a random chance. However, cross-gender communication takes longer and is faster paced as it seems that people tend to pay more attention when communicating with the opposite sex.

Recently, using the data we generated, Singla and Richardson further investigated the homophily within the Messenger network and found that people who communicate are also more likely to search the web for content on similar topics [Singla and Richardson, 2008].

9.7 The communication network

So far we have examined communication patterns based on pairwise communications. We now create a more general communication network from the data. Using this network, we can examine the typical *social distance* between people, *i.e.*, the number of links that separate a random pair of people. This analysis seeks to understand how many people can be reached within certain numbers of hops among people who communicate. Also, we test the transitivity of the network, *i.e.*, the degree at which pairs with a common friend tend to be connected.

We constructed a graph from the set of all two-user conversations, where each node corresponds to a person and there is an undirected edge between a pair of nodes if the users were engaged in an active

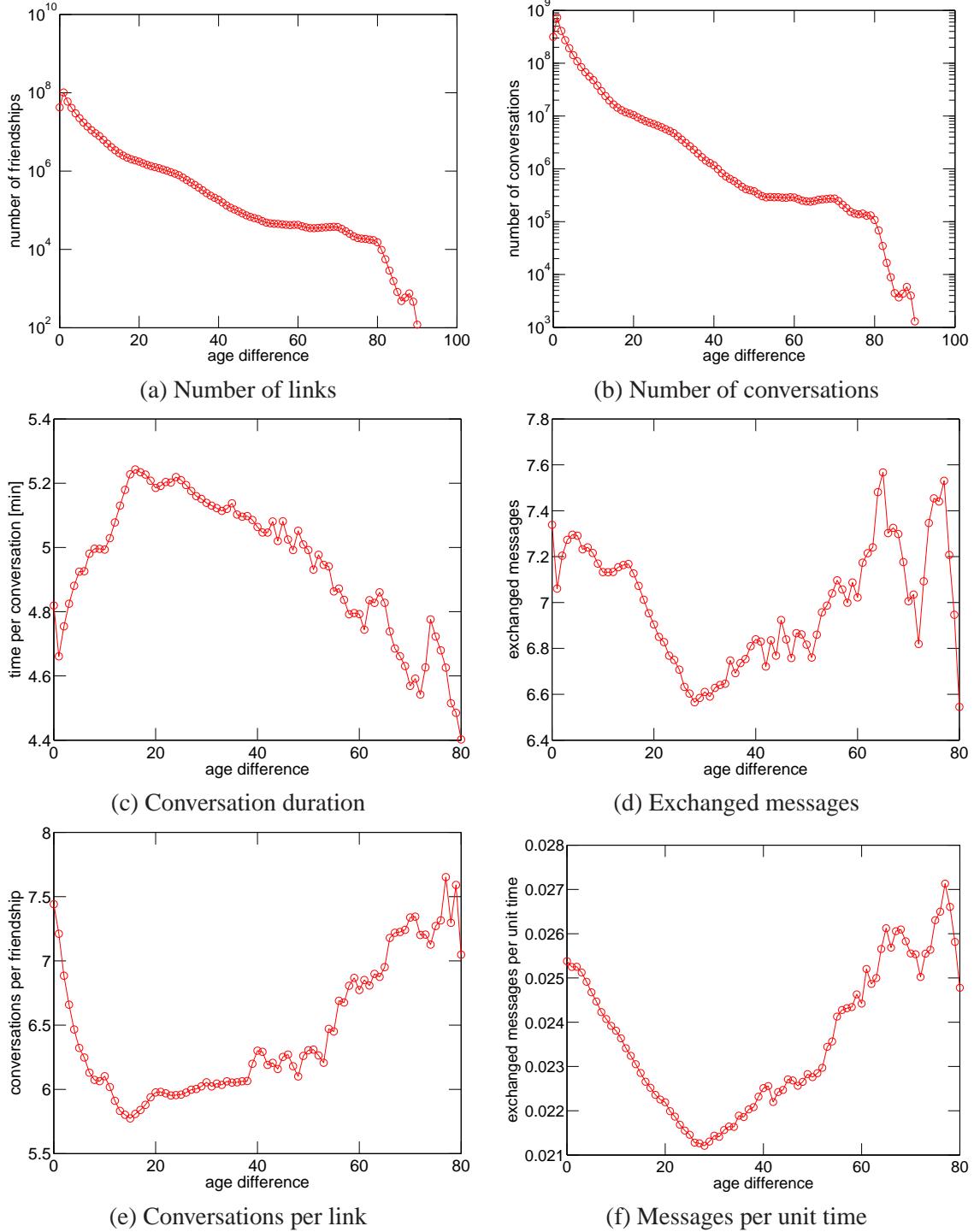


Figure 9.18: Communication characteristics with age difference between the users. (a) Number of links (pairs communicating) with the age difference. (b) Number of conversations. (c) Average conversation duration with the age difference. (d) Average number of exchanged messages per conversation as a function of the age difference between the users. (e) Number of conversations per link in the observation period with the age difference. (f) Number of exchanged messages per unit time as a function of age difference between the users.

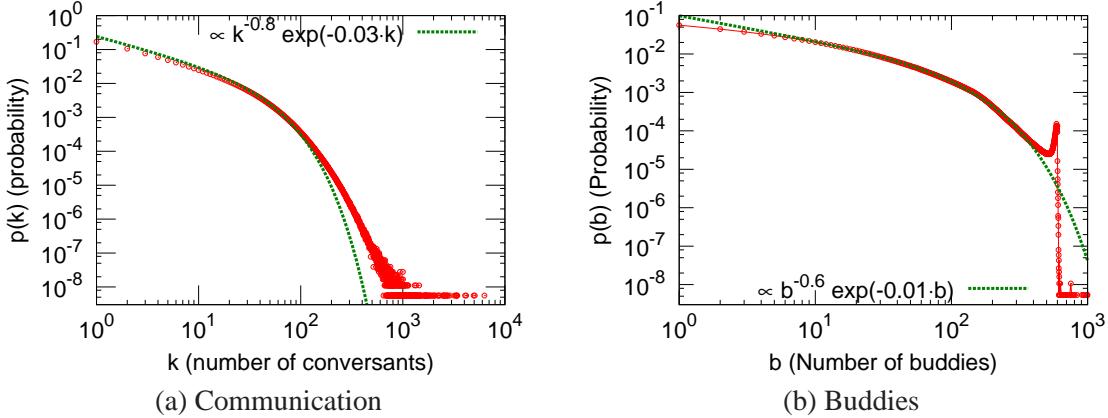


Figure 9.19: (a) Degree distribution of communication network (number of people with whom a person communicates). (b) Degree distribution of the buddy network (length of the contact list).

conversation during the observation period (users exchanged at least 1 message). The resulting network contains $N = 179,792,538$ nodes, and $E = 1,342,246,427$ edges. Note that this is not a *buddy network*; we only connect people who are buddies *and* have communicated during the observation period.

Figures 9.19–9.20 show the structural properties of the communication network. The network degree distribution shown in Figure 9.19(a) is heavy tailed but does not follow a power law distribution. Using maximum likelihood estimation, we fit a power law with exponential cutoff $p(d) \propto d^{-a} e^{-bd}$, where d denotes node degree. The fitted parameter values are $a = 0.8$ and $b = 0.03$. We found a strong cutoff parameter and low power law exponent, suggesting a distribution with high variance.

Figure 9.19(b) displays the degree distribution of a buddy graph. We did not have access to the full buddy network; we only had access to data on the length of the user contact list which allowed us to create the plot. We found a total of 9.1 billion buddy edges in the graph with 49 buddies per user. We fit the data with a power law distribution with exponential cutoff and identified parameters of $a = 0.6$ and $b = 0.01$. The power law exponent now is even smaller. This model described the data well. We note a spike at 600 which is the limit on the maximal number of buddies imposed by the Messenger software client. The maximal number of buddies was increased to 300 from 150 in March 2005, and was later raised to 600. With the data from June 2006, we see only the peak at 600, and could not identify bumps at the earlier constraints.

Social networks have been found to be highly transitive, *i.e.*, people with common friends tend to be friends themselves. The clustering coefficient [Watts and Strogatz, 1998] has been used as a measure of transitivity in the network. The measure is defined as the fraction of triangles around a node of degree d [Watts and Strogatz, 1998]. Figure 9.20(a) displays the clustering coefficient versus the degree of a node for Messenger. Previous results on measuring the web graph as well as theoretical analyses show that the clustering coefficient decays as d^{-1} (exponent -1) with node degree d [Ravasz and Barabási, 2003]. For the Messenger network, the clustering coefficient decays very slowly with exponent -0.37 with the degree of a node and the average clustering coefficient is 0.137. This result suggests that clustering in the Messenger network is much higher than expected—that people with common friends also tend to be connected. Figure 9.20(b) displays the distribution of the connected components in the network. The giant component contains 99.9% of the nodes in the network against a background of small components, and the distribution follows a power law.

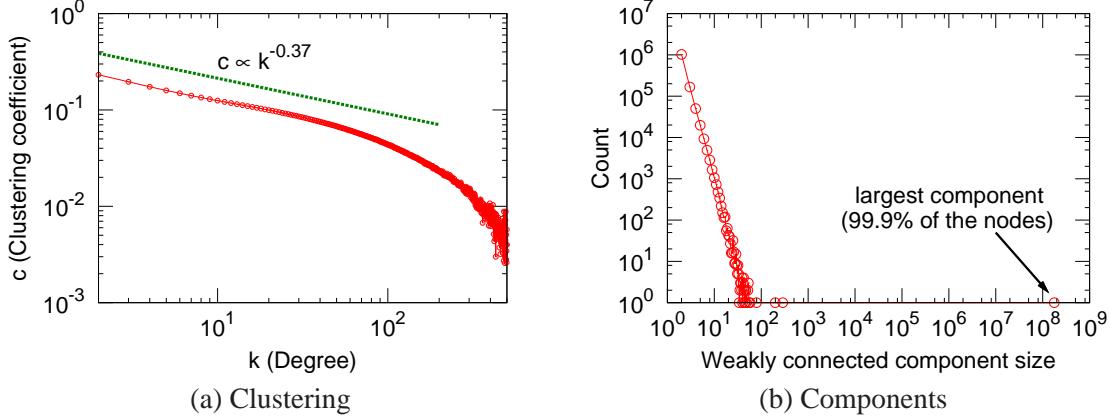


Figure 9.20: (a) Clustering coefficient; (b) distribution of connected components. 99.9% of the nodes belong to the largest connected component.

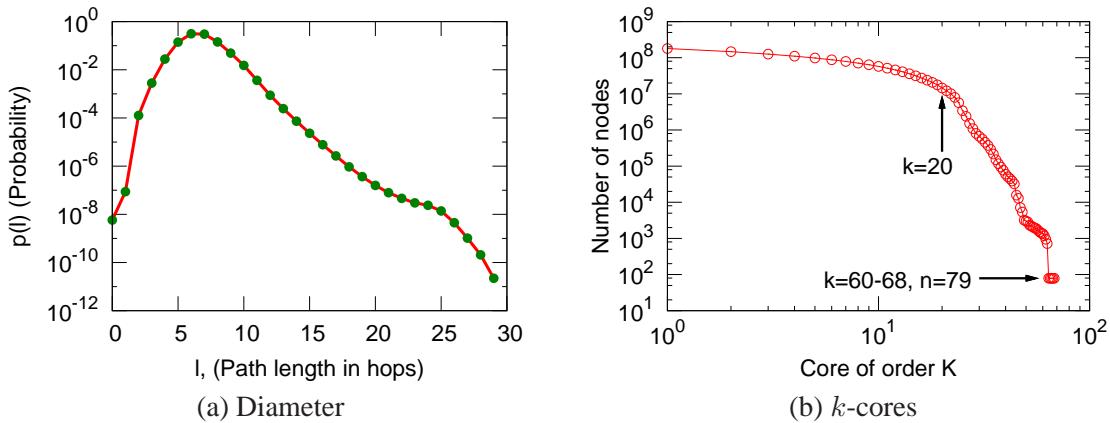


Figure 9.21: (a) Distribution over the shortest path lengths. Average shortest path has length 6.6, the distribution reaches the mode at 6 hops, and the 90% effective diameter is 7.8; (b) distribution of sizes of cores of order k .

9.7.1 How small is the small world?

Messenger data gives us a unique opportunity to study distances in the social network. To our knowledge, this is the first time a planetary-scale social network has been available to validate the well-known “6 degrees of separation” finding by Travers and Milgram [Milgram, 1967]. The earlier work employed a sample of 64 people and found that the average number of hops for a letter to travel from Nebraska to Boston was 6.2 (mode 5, median 5), which is popularly known as the “6 degrees of separation” among people. We used a population sample that is more than two million times larger than the group studied earlier and confirmed the classic finding.

Figure 9.21(a) displays the distribution over the shortest path lengths. To approximate the distribution of the distances, we randomly sampled 1000 nodes and calculated for each node the shortest paths to all other nodes. We found that the distribution of path lengths reaches the mode at 6 hops and has a median at 7. The average path length is 6.6. This result means that a random pair of nodes in the Messenger network

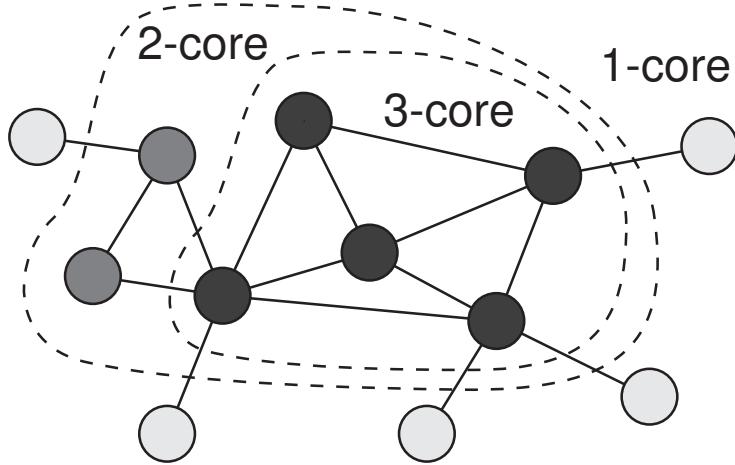


Figure 9.22: k -core decomposition of a small graph. Nodes contained in each closed line belong to a given k -core. Inside each k -core all nodes have degree larger than k (after removing all nodes with degree less than k).

is 6.6 hops apart on the average, which is half a link longer than the length measured by Milgram. The 90th percentile (effective diameter [Tauro et al., 2001]) of the distribution is 7.8. 48% of nodes can be reached within 6 hops and 78% within 7 hops. So, we might say that, via the lens provided on the world by Messenger, we find that there are about “7 degrees of separation” among people. We note that long paths, *i.e.*, nodes that are far apart, exist in the network; we found paths up to a length of 29 links.

It is an interesting question to hypothesize what could be the true degree of separation of the human race. At first sight one would think that if one could consider all the Earth’s population then diameter of such network would increase a bit. On the other hand we made observations in Chapter 3 from where we know that as networks grow they densify and the diameter shrinks, which would suggest that the true number of degrees of separation of human race is probably a bit smaller than what we found. On the other hand, our network is not complete. It is missing nodes and edges in particular geographic parts of the world. For example, as we saw in Figure 9.10 MSN population in Africa or South America is concentrated mostly on the coasts. This means the network has “holes” and one plausible hypothesis would be that the filling in these holes would further bring down the diameter and the average degree of separation.

9.7.2 Network cores

We further study connectivity of the communication network by examining the k -core decomposition of a network [Batagelj and Zaveršnik, 2002]. The concept of k -core is a generalization of the giant connected component. The k -core of a network is a set of vertices K , where each vertex in K has at least k edges to other vertices in K (see Figure 9.22). The distribution of k -core sizes gives us an idea of how quickly the network shrinks as we move towards the core.

The k -core of a graph can be obtained by deleting from the network all vertices of degree less than k . This process will decrease degrees of some non-deleted vertices, so more vertices will have degree less than k . We keep pruning vertices until all remaining vertices have degree of at least k . We call the remaining vertices a k -core.

Figure 9.21 plots the number of nodes in a core of order k . We note that the core sizes are remarkably stable up to a value of $k \approx 20$; the number of nodes in the core drops for only an order of magnitude. After $k > 20$, the core size rapidly drops. The central part of the communication network is composed of 79 nodes, where each of them has more than 68 edges inside the set. The structure of the Messenger communication network is quite different from the Internet graph; it has been observed [Alvarez-Hamelin et al., 2005] that the size of a k -core of the Internet decays as a power law with k . Here we see that the core sizes remains very stable up to a degree ≈ 20 , and only then start to rapidly decrease. This means that the nodes with degrees of less than 20 are on the fringe of the network, and that the core starts to rapidly decrease as nodes of degree 20 or more are deleted.

9.7.3 Strength of the ties

It has been observed by Albert et al. [Albert et al., 2000] that many real-world networks are robust to node-level changes or *attacks*. Researchers have showed that networks like the World Wide Web, Internet, and several social networks display a high degree of robustness to random node removals, *i.e.*, one has to remove many nodes chosen uniformly at random to make the network disconnected. On the contrary, targeted attacks are very effective. Removing a few high degree nodes can have a dramatic influence on the connectivity of a network.

Let us now study how the Messenger communication network is decomposed when “strong,” *i.e.*, heavily used, edges are removed from the network. We consider several different definitions of “heavily used,” and measure the types of edges that are most important for network connectivity. We note that a similar experiment was performed by Shi et al. [Shi et al., 2007] in the context of a small IM buddy network. The authors of the prior study took the number of common friends at the ends of an edge as a measure of the link strength. As the number of edges here is too large (1.3 billion) to remove edges one by one, we employed the following procedure: We order the nodes by decreasing value per a measure of the *intensity of engagement* of users; we then delete nodes associated with users in order of decreasing measure and we observe the evolution of the properties of the communication network as nodes are deleted.

We consider the following different measures of engagement:

- Average sent: The average number of sent messages per user’s conversation
- Average time: The average duration of user’s conversations
- Links: The number of links of a user (node degree), *i.e.*, number of different people he or she exchanged messages with
- Conversations: The total number of conversations of a user in the observation period
- Sent messages: The total number of sent messages by a user in the observation period
- Sent per unit time: The number of sent messages per unit time of a conversation
- Total time: The total conversation time of a user in the observation period

At each step of the experiment, we remove 10 million nodes in order of the specific measure of engagement being studied. We then determine the relative size of the largest connected component, *i.e.*, given the network at particular step, we find the fraction of the nodes belonging to the largest connected component of the network.

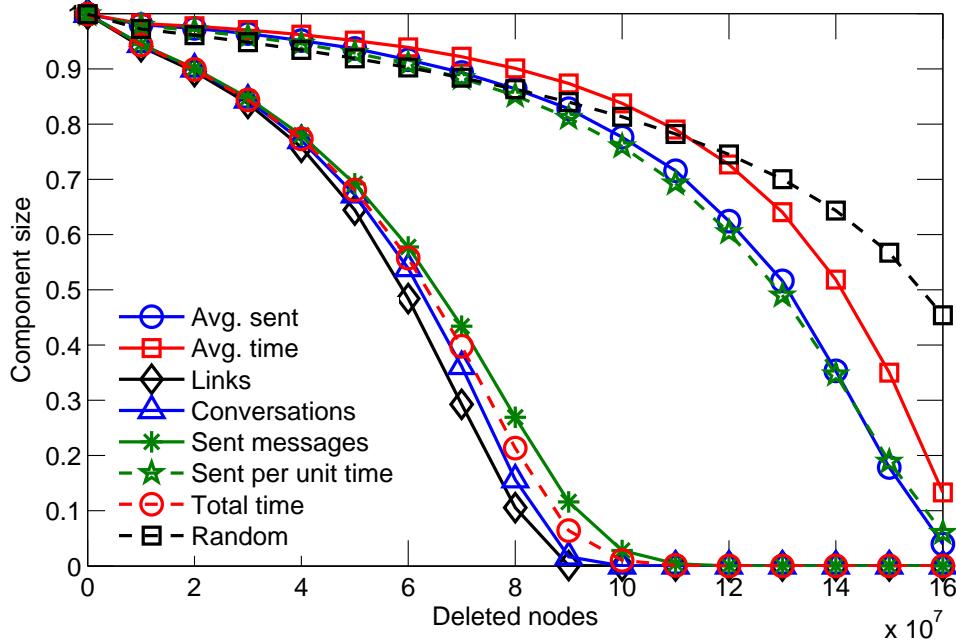


Figure 9.23: Relative size of the largest connected component as a function of number of nodes removed.

Figure 9.23 plots the evolution of the fraction of nodes in the largest connected component with the number of deleted nodes. We plot a separate curve for each of the seven different measures of engagement. For comparison, we also consider the random deletion of the nodes.

The decomposition procedure highlighted two types of dynamics of network change with node removal. The size of the largest component decreases rapidly when we use as measures of engagement the number of links, number of conversations, total conversation time, or number of sent messages. In contrast, the size of the largest component decreases very slowly when we use as a measure of engagement the average time per conversation, average number of sent messages, or number of sent messages per unit time. We were not surprised to find that the size of the largest component size decreases most rapidly when nodes are deleted in order of the decreasing number of links that they have, *i.e.*, the number of people with whom a user at a node communicates. Random ordering of the nodes shrinks the component at the slowest rate. After removing 160 million out of 180 million nodes with the random policy, the largest component still contains about half of the nodes. Surprisingly, when deleting up to 100 million nodes, the average time per conversation measure shrinks the component even more slowly than the random deletion policy.

Figure 9.24 displays plots of the number of removed edges from the network as nodes are deleted. Similar to the relationships in Figure 9.23, we found that deleting nodes by the inverse number of edges removes edges the fastest. As in Figure 9.24, the same group of node ordering criteria (number of conversations, total conversation time or number of sent messages) removes edges from the networks as fast as the number of links criteria. However, we find that random node removal removes edges in a linear manner. Edges are removed at a lower rate when deleting nodes by average time per conversation, average numbers of sent messages, or numbers of sent messages per unit time. We believe that these findings demonstrate that users with long conversations and many messages per conversation tend to have smaller degrees—even given the findings displayed in Figure 9.23, where we saw that removing these users is more effective

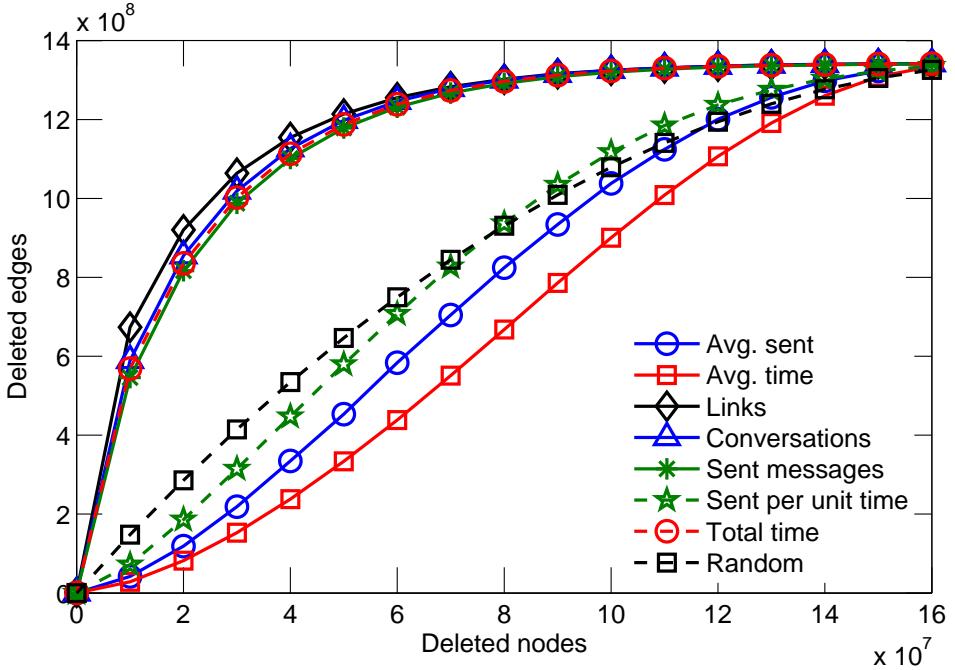


Figure 9.24: Number of removed edges as nodes are deleted by order of different measures of engagement.

for breaking the connectivity of the network than for random node deletion. Figure 9.24 also shows that using the average number of messages per conversation as a criterion removes edges in the slowest manner. We believe that this makes sense intuitively: If users invest similar amounts of time to interacting with others, then people with short conversations will tend to converse with more people in a given amount of time than users having long conversations.

9.8 Conclusion

We have reviewed a set of results stemming from the generation and analysis of an anonymized dataset representing the communication patterns of all people using a popular IM system. The methods and findings highlight the value of using a large IM network as a worldwide lens onto aggregate human behavior.

We described the creation of the dataset, capturing high-level communication activities and demographics in June 2006. The core dataset contains more than 30 billion conversations among 240 million people. We discussed the creation and analysis of a communication graph from the data containing 180 million nodes and 1.3 billion edges. The communication network is largest social network analyzed to date. The planetary-scale network allowed us to explore dependencies among user demographics, communication characteristics, and network structure. Working with such a massive dataset allowed us to test hypotheses such as the average chain of separation among people across the entire world.

We discovered that the graph is well connected, highly transitive, and robust. We reviewed the influence of multiple factors on communication frequency and duration. We found strong influences of homophily

in activities, where people with similar characteristics tend to communicate more, with the exception of gender, where we found that cross-gender conversations are both more frequent and of longer duration than conversations with users of the same reported gender. We also examined the path lengths and validated on a planetary scale earlier research that found “6 degrees of separation” among people.

We note that the sheer size of the data limits the kinds of analyses one can perform. In some cases, a smaller random sample may avoid the challenges with working with terabytes of data. However, it is known that sampling can corrupt the structural properties of networks, such as the degree distribution and the diameter of the graphs [Stumpf et al., 2005]. Thus, while sampling may be valuable for managing complexity of analyses, results on network properties with partial data sets may be rendered unreliable. Furthermore, we need to consider the full data set to reliably measure the patterns of age and distance homophily in communications.

In other directions of research with the dataset, we have pursued the use of machine learning and inference to learn predictive models that can forecast such properties as communication frequencies and durations of conversations among people as a function of the structural and demographic attributes of conversants. Our future directions for research include gaining an understanding of the dynamics of the structure of the communication network via a study of the evolution of the network over time.

We hope that our studies with Messenger data serves as an example of directions in social science research, highlighting how communication systems can provide insights about high-level patterns and relationships in human communications without making incursions into the privacy of individuals. We hope that this first effort to understand a social network on a genuinely planetary scale will embolden others to explore human behavior at large scales.

Chapter 10

Network community structure

How well do real networks partition into communities? What is a good way to measure and characterize presence or absence of community structure in networks? What are typical community sizes and typical community scores?

A large body of work has been devoted to identifying community structure in networks. A community is often thought of as a set of nodes that has more connections between its members than to the remainder of the network. In this chapter, we characterize as a function of size the statistical and structural properties of such sets of nodes. We define the *network community profile plot*, which characterizes the “best” possible community—according to the conductance measure—over a wide range of size scales, and we study over 100 large sparse real-world networks taken from a wide range of application domains. Our results suggest a significantly more refined picture of community structure in large real-world networks than has been appreciated previously.

Our most interesting finding is that in nearly every network dataset we examined, we observe tight but almost trivial communities at very small size scales, and at larger size scales, the best possible communities gradually “blend in” with the rest of the network and thus become less “community-like.” This behavior is not explained, even at a qualitative level, by most of the commonly-used network generation models. Moreover, this behavior is exactly the opposite of what one would expect based on experience with and intuition from expander graphs, from graphs that are well-embeddable in a low-dimensional structure, and from small social networks that have served as testbeds of community detection algorithms. We have found, however, that a generative model, in which new edges are added via an iterative Forest Fire burning process, is able to produce graphs exhibiting a network community structure similar to our observations.

10.1 Introduction

Defining and identifying communities or densely linked clusters in social and information networks, *i.e.*, in graphs where nodes represent underlying social entities and the edges represent interaction between pairs of nodes, has been studied in great detail. Most this research begins with the premise that a community should be thought of as a set of nodes that has more and/or better connections between its members than between members of that set and the remainder of the network. Here, we explore from a novel perspective

several questions related to identifying meaningful communities in large social and information networks. As we analyze large networks we are able to observe phenomena practically invisible in small networks and we come to several surprising conclusions that have implications for community detection in such networks.

Rather than define a procedure to extract a set of nodes from a graph and then attempt to interpret that set as a meaningful community, we will employ approximation algorithms for the graph partitioning problem in an attempt to characterize as a function of size the statistical and structural properties of partitions of graphs that could plausibly be interpreted as meaningful communities. In particular, we define the *network community profile plot*, which attempts to characterize the “best” possible community—according to the conductance measure—over a wide range of size scales. We study over 100 large sparse real-world networks taken from a wide range of application domains (ranging from traditional and on-line social networks, to technological and information networks and web graphs, and ranging in size from thousands of nodes up to tens of millions of nodes), and for each of these networks we compute a wide range of statistics, including “regularized” and “non-regularized” versions of the network community profile plot.

Our results suggest a significantly more refined picture of community structure in large networks than has been appreciated previously. Our observations agree with previous work on small networks, but we show that large networks have a very different structure. In particular, we observe tight communities that are barely connected to the rest of the network at very small size scales (up to ≈ 100 nodes); and communities of size scale beyond ≈ 100 nodes gradually “blend into” the expander-like core of the network and thus become less “community-like,” with a roughly inverse relationship between community size and optimal community quality. This observation agrees well with the so-called Dunbar number which gives a limit to the size of a well-functioning community.

This behavior is not explained, even at a qualitative level, by any of the commonly-used network generation models. Moreover, this behavior is exactly the opposite of what one would expect based on experience with and intuition from expander graphs, from graphs that are well-embeddable in a low-dimensional structure, and from small social networks that have served as testbeds of community detection algorithms. Certain aspects of it, *e.g.*, the existence of deep cuts or well-defined “communities” at small size scales and the non-existence of them at very large scales, are a consequence of the extreme sparsity of the networks, as we demonstrate by analyzing sparse random graph models. Other aspects of it, *e.g.*, the relatively gradual increase of the network community profile plot as a function of increasing size scale, depend in a subtle manner on the way in which local clustering information is propagated from smaller to larger size scales in the network. We have found that a generative graph model, in which new edges are added via an iterative Forest Fire burning process (we originally introduced it in Section 3.4), is able to produce graphs exhibiting a network community profile plot similar to what we observe in our network datasets.

10.1.1 Overview of our approach

Lots of effort has been devoted to the task of defining and identifying communities in networks. Most recent papers on the subject of community detection in large networks begin by noting that it is a matter of common experience that communities exist in such networks. These papers then note that, although there is no agreed-upon definition for a community, a community should be thought of as a set of nodes that has more and/or better connections between its members than between its members and the remainder of the network. These papers then apply a range of algorithmic techniques and intuitions to extract subsets

of nodes and then interpret these subsets as meaningful communities corresponding to some underlying “true” real-world communities. In this chapter, we explore from a novel perspective several questions related to identifying meaningful communities in large sparse networks, and we come to several striking conclusions that have implications for community detection and graph partitioning in such networks. We emphasize that, in contrast to most of the previous work on this subject, we look at very large networks of up to millions of nodes, and we observe very different phenomena than is seen in small commonly-analyzed networks.

At the risk of oversimplifying the large and often intricate body of work on community detection in complex networks, the following five-part story describes the general methodology:

- (1) Data are modeled by an “interaction graph.” In particular, part of the world gets mapped to a graph in which nodes represent entities and edges represent some type of interaction between pairs of those entities. For example, in a social network, nodes may represent individual people and edges may represent friendships, interactions or communication between pairs of those people.
- (2) The hypothesis is made that the world contains groups of entities that interact more strongly amongst themselves than with the outside world, and hence the interaction graph should contain sets of nodes, *i.e.*, communities, that have more and/or better-connected “internal edges” connecting members of the set than “cut edges” connecting the set to the rest of the world.
- (3) A objective function or metric is chosen to formalize this idea of groups with more intra-group than inter-group connectivity.
- (4) An algorithm is then selected to find sets of nodes that exactly or approximately optimize this or some other related metric. Sets of nodes that the algorithm finds are then called “clusters,” “communities,” “groups,” “classes,” or “modules”.
- (5) The clusters or communities or modules are evaluated in some way. For example, one may map the sets of nodes back to the real world to see whether they appear to make intuitive sense as a plausible “real” community. Alternatively, one may attempt to acquire some form of “ground truth,” in which case the set of nodes output by the algorithm may be compared with it.

With respect to points (1)–(4), we follow the usual path. In particular, we adopt points (1) and (2), and we then explore the consequence of making such a choice, *i.e.*, of making such an hypothesis and modeling assumption. For point (3), we choose a natural and widely-adopted notion of community goodness (community quality score) called *conductance*, which is also known as the normalized cut metric [Chung, 1997, Shi and Malik, 2000, Kannan et al., 2004]. Informally, the conductance of a set of nodes (defined and discussed in more detail in Section 10.2.3) is the ratio of the number of “cut” edges between that set and its complement divided by the number of “internal” edges inside that set. Thus, to be a good community, a set of nodes should have small conductance, *i.e.*, it should have many internal edges and few edges pointing to the rest of the network. Conductance is widely used to capture the intuition of a good community; it is a fundamental combinatorial quantity; and it has a very natural interpretation in terms of random walks on the interaction graph. Moreover, since there exist a rich suite of both theoretical and practical algorithms [Hendrickson and Leland, 1995, Spielman and Teng, 1996, Leighton and Rao, 1988, 1999, Arora et al., 2004b, Karypis and Kumar, 1998b,a, Zhao and Karypis, 2004, Dhillon et al., 2007], we can for point (4) compare and contrast several methods to approximately optimize it. To illustrate conductance, note that of the three 5-node sets A , B , and C illustrated in the graph in Figure 10.1, B has the best (the lowest) conductance and is thus the most community-like.

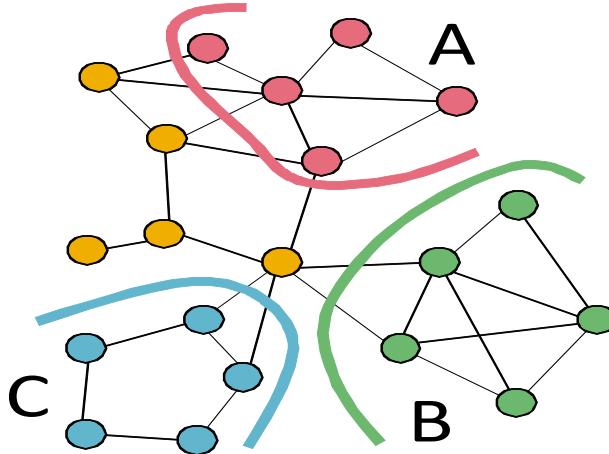


Figure 10.1: Network communities. Of the three 5-nodes sets that have been marked, *B* has the best (*i.e.*, the lowest) conductance, as it has the lowest ratio between the number of edges cut and the number of edges inside. So, set *B* is the best 5-node community or the most community-like set of 5 nodes in this particular network.

However, it is in point (5) that we deviate from previous work. Instead of focusing on individual groups of nodes and trying to interpret them as “real” communities, we investigate statistical properties of a large number of communities over a wide range of size scales in over 100 large sparse real-world social and information networks. We take a step back and ask questions such as: How well do real graphs split into communities? What is a good way to measure and characterize presence or absence of community structure in networks? What are typical community sizes and typical community qualities?

To address these and related questions, we introduce the concept of a *network community profile (NCP) plot* that we define and describe in more detail in Section 10.3.1. Intuitively, the network community profile plot measures the score of “best” community as a function of community size in a network. Formally, we define it as the conductance value of the minimum conductance set of cardinality k in the network, as a function of k . As defined, the NCP plot will be NP-hard to compute exactly, so operationally we will use several natural approximation algorithms for solving the Minimum Conductance Cut Problem in order to compute different approximations to it. By comparing and contrasting these plots for a large number of networks, and by computing other related structural properties, we obtain results that suggest a significantly more refined picture of the community structure in large real-world networks than has been appreciated previously.

We have gone to a great deal of effort to be confident that we are computing quantities fundamental to the networks we are considering, rather than artifacts of the approximation algorithms we employ. In particular:

- We use several classes of graph partitioning algorithms to probe the networks for sets of nodes that could plausibly be interpreted as communities. These algorithms, including flow-based methods, spectral methods, and hierarchical methods, have complementary strengths and weaknesses that are well understood both in theory and in practice. For example, flow-based methods are known to have difficulties with expanders [Leighton and Rao, 1988, 1999], and flow-based post-processing of other methods are known in practice to yield cuts with extremely good conductance values [Lang, 2004, Lang and Rao, 2004]. On the other hand, spectral methods are known to have difficulties when

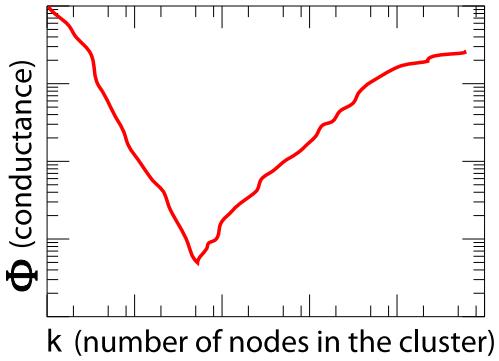
they confuse long paths with deep cuts [Spielman and Teng, 1996, Guattery and Miller, 1998], a consequence of which is that they may be viewed as computing a “regularized” approximation to the network community profile plot. (See Section 10.5 for a more detailed discussion of these and related issues.)

- We compute spectral-based lower bounds and also semidefinite-programming-based lower bounds for the conductance of our network datasets.
- We compute a wide range of other structural properties of the networks, *e.g.*, sizes, degree distributions, maximum and average diameters of the purported communities, internal versus external conductance values of the purported communities, etc.
- We recompute statistics on versions of the networks that have been modified in well-understood ways, *e.g.*, by removing small barely-connected sets of nodes or by randomizing the edges.
- We compare our results across not only over 100 large social and information networks, but also numerous commonly-studied small social networks, expanders, and low-dimensional manifold-like objects, and we compare our results on each network with what is known from the field from which the network is drawn. To our knowledge, this makes ours the most extensive such analysis of the community structure in large real-world social and information networks.
- We compare results with analytical and/or simulational results on a wide range of commonly and not-so-commonly used network generation models [Newman, 2003, Bollobas and Riordan, 2003, Barabási and Albert, 1999, Kumar et al., 2000, Ravasz and Barabási, 2003, Leskovec et al., 2005b, Flaxman et al., 2004, 2007].

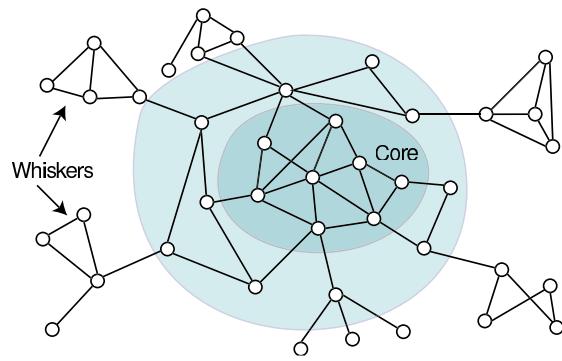
10.1.2 Summary of our results

Main Empirical Findings: Taken as a whole, the results we present in this chapter suggest a rather detailed and somewhat counterintuitive picture of the community structure in large social and information networks. Several qualitative properties of community structure, as revealed by the network community profile plot, are nearly universal:

- Up to a size scale, which empirically is roughly 100 nodes, there not only exist cuts with relatively good conductance, *i.e.*, good communities, but also the slope of the network community profile plot is generally sloping downward. This latter point suggests that smaller communities can be combined into meaningful larger communities, a phenomenon that we empirically observe in many cases.
- At the size scale of roughly 100 nodes, we often observe the global minimum of the network community profile plot; these are the “best” communities, according to the conductance measure, in the entire graph. These are, however, rather interestingly connected to the rest of the network; for example, in most cases, we observe empirically that they are a small set of nodes barely connected to the remainder of the network by just a *single* edge.
- Above the size scale of roughly 100 nodes, the network community profile plot gradually increases, and thus there is a nearly inverse relationship between community size and community quality. As a function of increasing size, the best possible communities become more and more “blended into” the remainder of the network. Intuitively, communities blend in with one another and gradually disappear as they grow larger. In particular, in many cases, larger communities can be broken



(a) Typical NCP plot



(b) Caricature of network structure

Figure 10.2: (a) Typical network community profile plot for a large social or information network: networks have better and better communities up to a size scale of ≈ 100 nodes, and after that size scale communities “blend-in” with the rest of the network (red curve). However, real networks still have more structure than their randomized (conditioned on the same degree distribution) counterparts (black curve). Even more surprisingly, if one allows for disconnected communities (blue curve), the community quality scores often get even better (even though such communities have no intuitive meaning). (b) Network structure for a large social or information network, as suggested by our empirical evaluations. See the text for more information on the “core” and “whiskers,” and note that the core in our real-world networks is actually extremely sparse.

into smaller and smaller pieces, often recursively, each of which is more community-like than the original supposed community.

- Even up to the largest size scales, we observe significantly more structure than would be seen, for example, in an expander-like random graph on the same degree sequence.

A schematic picture of a typical network community profile plot is illustrated in Figure 10.2(a). In red (labeled as “original network”), we plot community size vs. community quality score for the sets of nodes extracted from the original network. In black (rewired network), we plot the scores of communities extracted from a random network conditioned on the same degree distribution as the original network. This illustrates not only tight communities at very small scales, but also that at larger and larger size scales (the precise cutoff point for which is difficult to specify precisely) the best possible communities gradually “blend in” more and more with the rest of the network and thus gradually become less and less community-like. Eventually, even the existence of large well-defined communities is quite questionable if one models the world with an interaction graph, as in point (1) above, and if one also defines good communities as densely linked clusters that are weakly-connected to the outside, as in hypothesis (2) above. Finally, in blue (bag of whiskers), we also plot the scores of communities that are composed of disconnected pieces (found according to a procedure we describe in Section 10.4). This blue curve shows, perhaps somewhat surprisingly, that one can often obtain better community quality scores by combining unrelated disconnected pieces.

To understand the properties of generative models sufficient to reproduce the phenomena we have observed, we have examined in detail the structure of our social and information networks. Although nearly every network is an exception to any simple rule, we have observed that an “octopus” or “jellyfish” model [Chung and Lu, 2006a, Tauro et al., 2001, Siganos et al., 2006] provides a rough first approxima-

tion to structure of many of the networks we have examined. That is, most networks may be viewed as having a “core,” with no obvious underlying geometry and which contains a constant fraction of the nodes, and then there is a periphery consisting of a large number of relatively small “whiskers” that are only tenuously connected to the core. Figure 10.2(b) presents a caricature of this network structure. Of course, our network datasets are far from random in numerous ways—*e.g.*, they have higher edge density in the core; the small barely-connected whisker-like pieces are generally larger, denser, and more common than in corresponding random graphs; they have higher local clustering coefficients; and this local clustering information gets propagated globally into larger clusters or communities in a subtle and location-specific manner. More interestingly, as shown in Figure 10.13 in Section 10.4.4, the core itself consists of a nested core-periphery structure.

Main Modeling Results: The behavior that we observe is not reproduced, at even a qualitative level, by any of the commonly-used network generation models we have examined, including but not limited to preferential attachment models, copying models, small-world models, and hierarchical network models. Moreover, this behavior is qualitatively different than what is observed in networks with an underlying mesh-like or manifold-like geometry (which may not be surprising, but is significant insofar as these structures are often used as a scaffolding upon which to build other models), in networks that are good expanders (which may be surprising, since it is often observed that large social networks are expander-like), and in small social networks such as those used as testbeds for community detection algorithms (which may have implications for the applicability of these methods to detect large community-like structures in these networks). For the commonly-used network generation models, as well as for expander-like, low-dimensional, and small social networks, the network community profile plots are generally downward sloping or relatively flat.

Although it is well understood at a qualitative level that nodes that are “far apart” or “less alike” (in some sense) should be less likely to be connected in a generative model, understanding this point quantitatively so as to reproduce the empirically-observed relationship between small-scale and large-scale community structure turns out to be rather subtle. We can make the following observations:

- Very sparse random graph models with no underlying geometry have relatively deep cuts at small size scales, the best cuts at large size scales are very shallow, and there is a relatively abrupt transition in between. (This is shown pictorially in Figure 10.2(a) for a randomly rewired version of the original network.) This is a consequence of the extreme sparsity of the data: sufficiently dense random graphs do not have these small deep cuts; and the relatively deep cuts in sparse graphs are due to small tree-like pieces that are connected by a single edge to a core which is an extremely good expander.
- A Forest Fire generative model [Leskovec et al., 2005b, 2007b], in which edges are added in a manner that imitates a fire-spreading process, reproduces not only the deep cuts at small size scales and the absence of deep cuts at large size scales but other properties as well: the small barely connected pieces are significantly larger and denser than random; and for appropriate parameter settings the network community profile plot increases relatively gradually as the size of the communities increases.
- The details of the “forest fire” burning mechanism are crucial for reproducing how local clustering information gets propagated to larger size scales in the network, and those details shed light on the failures of commonly-used network generation models. In the Forest Fire Model, a new node selects a “seed” node and links to it. Then with some probability it “burns” or adds an edge to the each of the seed’s neighbors, and so on, recursively. Although there are elements of a “preferential

attachment” and also a “copying” flavor to this mechanism, two factors are particularly important: first is the local (in a graph sense, as there is no underlying geometry in the model) manner in which the edges are added; and second is that the number of edges that a new node can add can vary widely, depending on the local structure around the seed node. Depending on the neighborhood structure around the seed, small fires will keep the community well-separated from the network, but occasional large fires will connect the community around the seed node to the rest of the network and make it blend into the network core.

Thus, intuitively, the structure of the whiskers (components connected to the rest of the graph via a single edge) are responsible for the downward part of the network community profile plot, while the core of the network and the manner in which the whiskers root themselves to the core helps to determine the upward part of the network community profile plot. Due to local clustering effects, whiskers in real networks are larger and give deeper cuts than whiskers in corresponding randomized graphs, fluctuations in the core are larger and deeper than in corresponding randomized graphs, and thus the network community profile plot increases more gradually and levels off to a conductance value well below the value for a corresponding rewired network.

Main Methodological Contributions: To obtain these and other conclusions, we have employed approximation algorithms for graph partitioning to investigate structural properties of our network datasets. Briefly, we have done the following:

- We have used what we refer to as Metis+MQI, which consists of using the popular graph partitioning package Metis [Karypis and Kumar, 1998b] followed by a flow-based MQI post-processing [Lang and Rao, 2004]. With this procedure, we obtain sets of nodes that have really good conductance scores. (As we will later see mostly at the expense of cluster compactness.) This method heavily optimizes the conductance and does not consider the internal cluster structure. So, at very small size scales, these sets of nodes could plausibly be interpreted as good communities, but at larger size scales, we often obtain tenuously-connected (and in some cases unions of disconnected) pieces, which perhaps do not correspond to intuitive communities.
- Thus, we have also used the Local Spectral method of Anderson, Chung, and Lang [Andersen et al., 2006] to obtain sets of nodes with good conductance value that are “compact” or more “regularized” than those pieces returned by Metis+MQI. Since spectral methods confuse long paths with deep cuts [Spielman and Teng, 1996, Guattery and Miller, 1998], empirically we obtain sets of nodes that have worse conductance scores than sets returned by Metis+MQI, but which are “tighter” and more “community-like.” For example, at small size scales the sets of nodes returned by the Local Spectral Algorithm agrees with the output of Metis+MQI, but at larger cluster sizes this algorithm returns sets of nodes with substantially smaller diameter and average diameter. Such clusters are more compact and thus seem plausibly more community-like.

We have also used what we call the Bag-of-Whiskers Heuristic to identify small barely connected sets of nodes that exert a surprisingly large influence on the network community profile plot.

Both Metis+MQI and the Local Spectral Algorithm scale well and thus either may be used to obtain sets of nodes from very large graphs. For many of the small to medium-sized networks, we have checked our results by applying one or more other spectral, flow-based, or heuristic algorithms, although these do not scale as well to very large graphs. Finally, for some of our smaller network datasets, we have computed spectral-based and semidefinite-programming-based lower bounds, and the results are consistent with the conclusions we have drawn.

Broader implications: Our observation that, independently of the network size, compact communities exist only up to a size scale of around 100 nodes agrees well with the “Dunbar number” [Dunbar, 1998], which predicts that roughly 150 individuals is the upper limit on the size of a well-functioning human community. Moreover, we should emphasize that our results do not disagree with the literature at small sizes scales. One reason for the difference in our findings is that previous studies mainly focused on small networks, which are simply not large enough for the clusters to gradually blend into one another as one looks at larger size scales. In order to make our observations, one needs to look at large number (due to the complex noise properties of real graphs) of large networks. It is only when Dunbar’s limit is exceeded by several orders of magnitude that it is relatively easy to observe large communities blurring together and eventually vanishing. A second reason for the difference is that previous work did not measure and examine the *network community profile* of cluster size vs. cluster quality. Finally, we should note that our explanation also aligns well with the *common bond* vs. *common identity* theory of group attachment [Ren et al., 2007] from social psychology, where it has been noted that bond communities tend to be smaller and more cohesive [Back, 1951], as they are based on interpersonal ties, while identity communities are focused around common theme or interest. We discuss these implications and connections further in Section 10.7.

10.1.3 Outline of the chapter

The rest of the chapter is organized as follows. In Section 10.2 we describe some useful background, including a brief description of the network datasets we have analyzed. Then, in Section 10.3 we present our main results on the properties of the network community profile plot for our network datasets. We place an emphasis on how the phenomena we observe in large social and information networks are qualitatively different than what one would expect based on intuition from and experience with expander-like graphs, low-dimensional networks, and commonly-studied small social networks. Then, in Sections 10.4 and 10.5, we summarize the results of additional empirical evaluations. In particular, in Section 10.4, we describe some of the observations we have made in an effort to understand what structural properties of these large networks are responsible for the phenomena we observe; and in Section 10.5, we describe some of the results of probing the networks with different approximation algorithms in an effort to be confident that the phenomena we observed really are properties of the networks we study, rather than artifactual properties of the algorithms we chose to use to study those networks. We follow this in Section 10.6 with a discussion of complex network generation models. We observe that the commonly-used network generation models fail to reproduce the counterintuitive phenomena we observe. We also notice that very sparse random networks reproduce certain aspects of the phenomena, and that a generative model based on an iterative “forest fire” burning mechanism reproduces very well the qualitative properties of the phenomena we observe. Finally, in Section 10.7 we provide a discussion of our results in a broader context, and in Section 10.8 we present a brief conclusion.

10.2 Background on communities and overview of our methods

In this section, we will provide background on our data and methods. We start in Section 10.2.1 with a description of the network datasets we will analyze. Then, in Section 10.2.2, we review related community detection and graph clustering ideas. Finally, in Section 10.2.3, we provide a brief description of approximation algorithms that we will use. There exist a large number of reviews on topics related

to those discussed in this chapter. For example, see the reviews on community identification [Newman, 2004, Danon et al., 2005], data clustering [Jain et al., 1999], graph and spectral clustering [Gaertler, 2005, von Luxburg, 2006, Schaeffer, 2007], graph and heavy-tailed data analysis [Chakrabarti and Faloutsos, 2006, Newman, 2005, Clauset et al., 2007], surveys on various aspects of complex networks [Newman, 2003, Albert and Barabási, 2002, Dorogovtsev and Mendes, 2002a, Bollobas and Riordan, 2003, Li et al., 2005, da F. Costa et al., 2007, Boccaletti et al., 2006], the monographs on spectral graph theory and complex networks [Chung, 1997, Chung and Lu, 2006a], and the book on social network analysis methods and applications [Wasserman and Faust, 1994]. See Section 10.7 for a more detailed discussion of the relationship of our work with some of this prior work.

10.2.1 Social and information network datasets we analyze

We have examined a large number of real-world complex networks. See Tables A.2, A.3, and A.4 for a summary. For convenience, we have organized the networks into the following categories: Social networks; Information/citation networks; Collaboration networks; Web graphs; Internet networks; Bipartite affiliation networks; Biological networks; Low-dimensional networks; IMDB networks; and Amazon networks. We have also examined numerous small social networks that have been used as a testbed for community detection algorithms (*e.g.*, Zachary’s karate club [Zachary, 1977, Network data, 2007], interactions between dolphins [Lusseau et al., 2003, Network data, 2007], interactions between monks [Sampson, 1968, Network data, 2007], Newman’s network science network [Newman, 2006a, Network data, 2007], etc.), numerous simple network models in which by design there is an underlying geometry (*e.g.*, power grid and road networks [Watts and Strogatz, 1998], simple meshes, low-dimensional manifolds including graphs corresponding to the well-studied “swiss roll” data set [Tenenbaum et al., 2000], a geometric preferential attachment model [Flaxman et al., 2004, 2007], etc.), several networks that are very good expanders, and many simulated networks generated by commonly-used network generation models(*e.g.*, preferential attachment models [Newman, 2003], copying models [Kumar et al., 2000], hierarchical models [Ravasz and Barabási, 2003], etc.).

Social networks: The class of social networks in Table A.2 is particularly diverse and interesting. It includes several large on-line social networks: a network of professional contacts from LinkedIn (LINKEDIN); a friendship network of a LiveJournal blogging community (LIVEJOURNAL01); and a who-trusts-whom network of Epinions (EPINIONS). It also includes an email network from Enron (EMAIL-ENRON) and from a large European research organization. For the latter we generated three networks: EMAIL-INSIDE uses only the communication inside organization; EMAIL-INOUT also adds external email addresses where email has been sent both way; and EMAIL-ALL adds all communication inside the organization and to the outside world. Also included in the class of social networks are networks that are not the central focus of the websites from which they come, but which instead serve as a tool for people to share information more easily. For example, we have: the networks of a social bookmarking site Delicious (DELICIOUS); a Flickr photo sharing website (FLICKR); and a network from Yahoo! Answers question answering website (ANSWERS). In all these networks, a node refers to an individual and an edge is used to indicate that means that one person has some sort of interaction with another person, *e.g.*, one person subscribes to their neighbor’s bookmarks or photos, or answers their questions.

Information and citation networks: The class of Information/citation networks contains several different citation networks. It contains two citation networks of physics papers on arxiv.org, (CIT-HEP-TH and CIT-HEP-PH), and a network of citations of US patents (CIT-PATENTS). (These paper-to-paper citation networks are to be distinguished from scientific collaboration networks and author-to-paper bipartite

networks, as described below.) It also contains two types of blog citation networks. In the so-called post networks, nodes are posts and edges represent hyperlinks between blog posts (POST-NAT05-6M and POST-NAT06ALL). On the other hand, the so-called blog network is the blog-level-aggregation of the same data, *i.e.*, there is a link between two blogs if there is a post in first that links the post in a second blog (BLOG-NAT05-6M and BLOG-NAT06ALL).

Collaboration networks: The class of collaboration networks contain academic collaboration (*i.e.*, co-authorship) networks between physicists from various categories in `arxiv.org` (CA-ASTRO-PH, etc.) and between authors in computer science (CA-DBLP). It also contains a network of collaborations between pairs of actors in IMDB (ATA-IMDB), *i.e.*, there is an edge connecting a pair of actors if they appeared in the same movie. (Again, this should be distinguished from actor-to-movie bipartite networks, as described below.)

Web graphs: The class of Web graph networks includes four different web-graphs in which nodes represent web-pages and edges represent hyperlinks between those pages. Networks were obtained from Google (WEB-GOOGLE), the University of Notre Dame (WEB-NOTREDAME), TREC (WEB-TREC), and Stanford University (WEB-BERKSTAN). The class of Internet networks consists of various autonomous systems networks obtained at different sources, as well as a Gnutella and eDonkey peer-to-peer file sharing networks.

Bipartite networks: The class of Bipartite networks is particularly diverse and includes: authors-to-papers graphs from both computer science (ATP-DBLP) and physics (ATP-ASTRO-PH, etc.); a network representing users and the URLs they visited (CLICKSTREAM); a network representing users and the movies they rated (NETFLIX); and a users-to-queries network representing query terms that users typed into a search engine (QUERYTERMS). (We also have analyzed several bipartite actors-to-movies networks extracted from the IMDB database, which we have listed separately below.)

Biological networks: The class of Biological networks include protein-protein interaction networks of yeast obtained from various sources.

Low dimensional grid-like networks: The class of Low-dimensional networks consists of graphs constructed from road (ROAD-CA, etc.) or power grid (POWERGRID) connections and as such might be expected to “live” on a two-dimensional surface in a way that all of the other networks do not. We also added a “swiss roll” network, a 2-dimensional manifold embedded in 3-dimensions, and a “Faces” dataset where each point is an 64 by 64 gray-scale image of a face (embedded in 4,096 dimensional space) and where we connected the faces that were most similar (using the Euclidean distance).

IMDB, Yahoo! Answers and Amazon networks: Finally, we have networks from IMDB, Amazon, and Yahoo! Answers, and for each of these we have separately analyzed subnetworks. The IMDB networks consist of actor-to-movie links, and we include the full network as well as subnetworks associated with individual countries based on the country of production. For the Amazon networks, recall that Amazon sells a variety of products, and for each item A one may compile the list the up to ten other items most frequently purchased by buyers of A . This information can be presented as a directed network in which vertices represent items and there is a edge from item A to another item B if B was frequently purchased by buyers of A . We consider the network as undirected. We use five networks from a study of Clauset *et al.* [Clauset et al., 2004], and two networks from the viral marketing study from Leskovec *et al.* [Leskovec et al., 2007a]. Finally, for the Yahoo! Answers networks, we observe several deep cuts at large size scales, and so in addition the full network, we analyze the top six most well-connected subnetworks.

In addition to providing a brief description of the network, Tables A.2, A.3 and A.4 show the number of nodes and edges in each network, as well as other statistics which will be described in Section 10.4.1. (In all cases, we consider the network as undirected, and we extract and analyze the largest connected component.) The sizes of these networks range from about 5,000 nodes up to nearly 14 million nodes, and from about 6,000 edges up to more than 100 million edges. All of the networks are quite sparse—their densities range from an average degree of about 2.5 for the blog post network, up to an average degree of about 400 in the network of movie ratings from Netflix, and most of the other networks, including the purely social networks, have average degree around 10 (median average degree of 6). In many cases, we examined several versions of a given network. For example, we considered the entire IMDB actor-to-movie network, as well as sub-pieces of it corresponding to different language and country groups. Detailed statistics for all these networks are presented in Tables A.2, A.3 and A.4 and are described in Section 10.4. In total, we have examined over 100 large real-world social and information networks, making this, to our knowledge, the largest and most comprehensive study of such networks.

10.2.2 Clusters and communities in networks

Hierarchical clustering or linkage clustering is a common approach to community identification in social sciences [Wasserman and Faust, 1994], but it has also found application more generally [Hopcroft et al., 2004, Girvan and Newman, 2002]. In this procedure, one first defines a distance metric between pairs of nodes and then produces a tree (in either a bottom-up or a top-down manner) describing how nodes group into communities and how these group further into super-communities. A quite different approach that has received a great deal of attention (and that will be central to our analysis) is based on ideas from *graph partitioning* [Schaeffer, 2007, Brandes et al., 2007]. In this case, the network is modeled as simple undirected graph, where nodes and edges have no attributes, and a partition of the graph is determined by optimizing a merit function. The graph partitioning problem is find some number k groups of nodes, generally with roughly equal size, such that the number of edges between the groups, perhaps normalized in some way, is minimized.

Let $G = (V, E)$ denote a graph, then the *conductance* ϕ of a set of nodes $S \subset V$, (where S is assumed to contain no more than half of all the nodes), is defined as follows. Let v be the sum of degrees of nodes in S , and let s be the number of edges with one endpoint in S and one endpoint in \bar{S} , where \bar{S} denotes the complement of S . Then, the conductance of S is $\phi = s/v$, or equivalently $\phi = s/(s + 2e)$, where e is the number of edges with both endpoints in S . More formally:

Definition 10.2.1. Given a graph G with adjacency matrix A the conductance of a set of nodes S is defined as:

$$\phi(S) = \frac{\sum_{i \in S, j \notin S} A_{ij}}{\min\{A(S), A(\bar{S})\}}, \quad (10.6)$$

where $A(S) = \sum_{i \in S} \sum_{j \in V} A_{ij}$, or equivalently $A(S) = \sum_{i \in S} d(i)$, where $d(i)$ is a degree of node i in G .

Moreover, in this case, the conductance of the graph G is:

$$\phi_G = \min_{S \subset V} \phi(S). \quad (10.7)$$

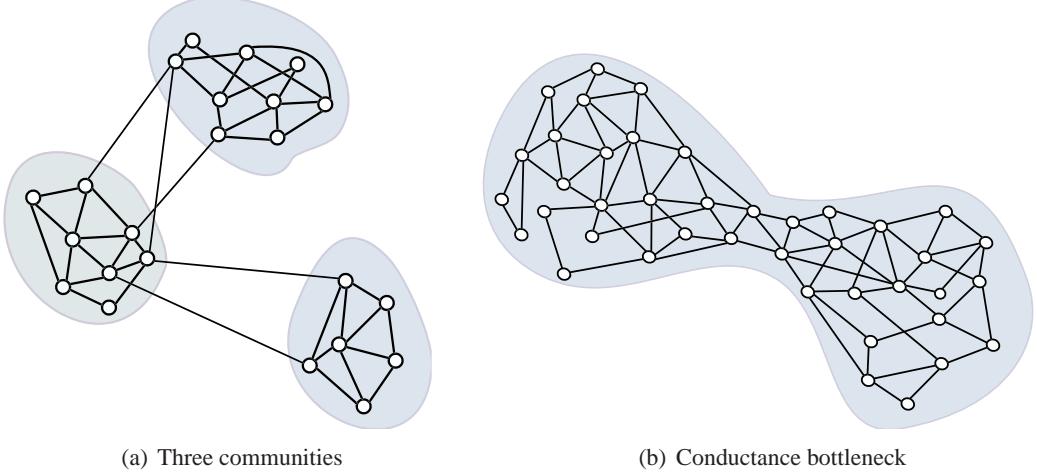


Figure 10.3: (a) Caricature of the traditional view of communities as being sets of nodes with more and/or better intra-connections than inter-connections. (b) A graph with its minimum conductance bottleneck illustrated.

Thus, the conductance of a set provides a measure for the quality of the cut (S, \bar{S}) , or relatedly the goodness of a community S .¹

Indeed, it is often noted that communities should be thought of as sets of nodes with more and/or better intra-connections than inter-connections; see Figure 10.3 for an illustration. When interested in detecting communities and evaluating their quality, we prefer sets with small conductances, *i.e.*, sets that are densely linked inside and sparsely linked to the outside. Although numerous measures have been proposed for how community-like is a set of nodes, it is commonly noted—*e.g.*, see Shi and Malik [Shi and Malik, 2000] and Kannan, Vempala, and Vetta [Kannan et al., 2004]—that conductance captures the “gestalt” notion of clustering [Zahn, 1971], and as such it has been widely-used for graph clustering and community detection [Gaertler, 2005, von Luxburg, 2006, Schaeffer, 2007].

There are many other density-based measures that have been used to partition a graph into a set of communities [Gaertler, 2005, von Luxburg, 2006, Schaeffer, 2007]. One that deserves particular mention is modularity [Newman and Girvan, 2004, Newman, 2006b]. For a given partition of a network into a set of communities, modularity measures the number of within-community edges, relative to a null model that is usually taken to be a random graph with the same degree distribution. Thus, modularity was originally introduced and it typically used to measure the strength or quality of a particular partition of a network. We, however, are interested in a quite different question than those that motivated the introduction of modularity. Rather than seeking to partition a graph into the “best” possible partition of communities, we would like to know how good is a particular element of that partition, *i.e.*, how community-like are the best possible communities that modularity or any other merit function can hope to find, in particular as a function of the size of that partition.

¹ Throughout this chapter we consistently use shorthand phrases like “this piece has good conductance” to mean “this piece is separated from the rest of the graph by a low-conductance cut.”

10.2.3 Approximation algorithms for finding low-conductance cuts

In addition to capturing very well our intuitive notion of what it means for a set of nodes to be a good community, the use of conductance as an objective function has an added benefit: there exists an extensive theoretical and practical literature on methods for approximately optimizing it. (Finding cuts with exactly minimal conductance is NP-hard.) In particular, the theory literature contains several algorithms with provable approximation performance guarantees.

First, there is the spectral method, which uses an eigenvector of the graph's Laplacian matrix to find a cut whose conductance is no bigger than ϕ if the graph actually contains a cut with conductance $O(\phi^2)$ [Cheeger, 1969, Donath and Hoffman, 1972, Fiedler, 1973, Mohar, 1991, Chung, 1997]. The spectral method also produces lower bounds which can show that the solution for a given graph is closer to optimal than promised by the worst-case guarantee. Second, there is an algorithm that uses multi-commodity flow to find a cut whose conductance is within an $O(\log n)$ factor of optimal [Leighton and Rao, 1988, 1999]. Spectral and multi-commodity flow based methods are complementary in that the worst-case $O(\log n)$ approximation factor is obtained for flow-based methods on expander graphs [Leighton and Rao, 1988, 1999], a class of graphs which does not cause problems for spectral methods, whereas spectral methods can confuse long path with deep cuts [Guattery and Miller, 1998, Spielman and Teng, 1996], a difference that does not cause problems for flow-based methods. Third, and very recently, there exists an algorithm that uses semidefinite programming to find a solution that is within $O(\sqrt{\log n})$ of optimal [Arora et al., 2004b]. This paper sparked a flurry of theoretical research on a family of closely related algorithms including [Arora et al., 2004a, Khandekar et al., 2006, Arora and Kale, 2007], all of which can be informally described as combinations of spectral and flow-based techniques which exploit their complementary strengths. However, none of those algorithms are currently practical enough to use in our study.

Of the above three theoretical algorithms, the spectral method is by far the most practical. Also very common are recursive bisection heuristics: recursively divide the graph into two groups, and then further subdivide the new groups until the desired number of clusters groups is achieved. This may be combined with local improvement methods like the Kernighan-Lin and Fiduccia-Mattheyses procedures [Kernighan and Lin, 1970, Fiduccia and Mattheyses, 1982], which are fast and can climb out of some local minima. The latter was combined with a multi-resolution framework to create Metis [Karypis and Kumar, 1998b,a], a very fast program intended to split mesh-like graphs into equal sized pieces. The authors of Metis later created Cluto [Zhao and Karypis, 2004], which is better tuned for clustering-type tasks. Finally we mention Graclus [Dhillon et al., 2007], which uses multi-resolution techniques and kernel k -means to optimize a metric that is closely related to conductance.

While the preceding were all approximate algorithms for finding the lowest conductance cut in a whole graph, we now mention MQI [Gallo et al., 1989, Lang and Rao, 2004], an *exact* algorithm for the slightly different problem of finding the lowest conductance cut in *half* of a graph. This algorithm can be combined with a good method for initially splitting the graph into two pieces (such as Metis or the Spectral method) to obtain a surprisingly strong heuristic method for finding low conductance cuts in the whole graph [Lang and Rao, 2004]. The exactness of the second optimization step frequently results in cuts with extremely low conductance scores, as will be visible in many of our plots. MQI can be implemented by solving single parametric max flow problems, or sequences of ordinary max flow problems. Parametric max flow (with MQI described as one of the applications) was introduced by [Gallo et al., 1989], and recent empirical work is described in [Babenko et al., 2007], but currently there is no publicly available code that scales to the sizes we need. Ordinary max flow is a very thoroughly studied problem.

Currently, the best theoretical time bounds are [Goldberg and Rao, 1998], the most practical algorithm is [Goldberg and Tarjan, 1988], while the best implementation is `hi_pr` by [Cherkassky and Goldberg, 1995]. Since Metis+MQI using the `hi_pr` code is very fast and scalable, while the method empirically seems to usually find the lowest or nearly lowest conductance cuts in a wide variety of graphs, we have used it extensively in this study.

We will also extensively use Local Spectral Algorithm of Andersen, Chung, and Lang [Andersen et al., 2006] to find node sets of low conductance, *i.e.*, good communities, around a seed node. This algorithm is also very fast, and it can be successfully applied to very large graphs to obtain more “well-rounded”, “compact,” or “evenly-connected” communities than those returned by Meits+MQI. The latter observation (described in more detail in Section 10.5) is since local spectral methods also confuse long paths (which tend to occur in our very sparse network datasets) with deep cuts. This algorithm takes as input two parameters—the seed node and a parameter ϵ that intuitively controls the locality of the computation—and it outputs a set of nodes. Local spectral methods were introduced by Spielman and Teng [Spielman and Teng, 2004, Andersen et al., 2006], and they have roughly the same kind of quadratic approximation guarantees as the global spectral method, but they have computational cost is proportional to the size of the obtained piece [Chung, 2007a,c,b].

10.3 The Network Community Profile Plot (NCP plot)

In this section, we discuss the *network community profile plot* (NCP plot), which measures the quality of network communities at different size scales. We start in Section 10.3.1 by introducing it. Then, in Section 10.3.2, we present the NCP plot for several examples of networks which inform peoples’ intuition and for which the NCP plot behaves in a characteristic manner. Then, in Sections 10.3.3 and 10.3.4 we present the NCP plot for a wide range of large real world social and information networks. We will see that in such networks the NCP plot behaves in a qualitatively different manner.

10.3.1 Definitions for the network community profile plot

In order to more finely resolve community structure in large networks, we introduce the *network community profile plot* (NCP plot). Intuitively, the NCP plot measures the quality of the best possible community in a large network, as a function of the community size. Formally, we may define it as the conductance value of the best conductance set of cardinality k in the entire network, as a function of k .

Definition 10.3.1. *Given a graph G with adjacency matrix A , the network community profile plot (NCP plot) plots $\Phi(k)$ as a function of k , where*

$$\Phi(k) = \min_{S \subset V, |S|=k} \phi(S), \quad (10.8)$$

where $|S|$ denotes the cardinality of the set S , and where the conductance $\phi(S)$ of S is given by equation (10.6).

Since this quantity is intractable to compute, we will employ well-studied approximation algorithms for the Minimum Conductance Cut Problem to approximate it. In particular, operationally we will use several natural heuristics based on approximation algorithms to do graph partitioning in order to compute different approximations to the NCP plot. Although other procedures will be described in

Section 10.5, we will primarily employ two procedures. First, Metis+MQI, *i.e.*, the graph partitioning package Metis [Karypis and Kumar, 1998b] followed by the flow-based post-processing procedure MQI [Lang and Rao, 2004]; this procedure returns sets that have very good conductance values. Second, the Local Spectral Algorithm of Andersen, Chung, and Lang [Andersen et al., 2006]; this procedure returns sets that are somewhat more “compact” or “smoothed” or “regularized,” but that often have somewhat worse conductance values.

Just as the conductance of a set of nodes provides a quality measure of that set as a community, the shape of the NCP plot provides insight into the community structure of a graph as a whole. For example, the magnitude of the conductance tells us how well clusters of different sizes are separated from the rest of the network. One might hope to obtain some sort of “smoothed” measure of the notion of the best community of size k (*e.g.*, by considering an average of the conductance value over all sets of a given size or by considering a smoothed extremal statistic such as a 95-th percentile) rather than conductance of the best set of that size. We have not defined such a measure since there is no obvious way to average over all subsets of size k and obtain a meaningful approximation to the minimum. On the other hand, our approximation algorithm methodology implicitly incorporates such an effect. Although Metis+MQI finds sets of nodes with extremely good conductance value, empirically we observe that they often have little or no internal structure—they can even be disconnected. On the other hand, since spectral methods in general tend to confuse long paths with deep cuts [Spielman and Teng, 1996, Guattery and Miller, 1998], the Local Spectral Algorithm finds sets that are “tighter” and more “well-rounded” and thus in many ways more community-like. (See Sections 10.2.3 and 10.5 for details on these algorithmic issues and interpretations.)

10.3.2 NCP plots for small social networks, expander and low-dimensional graphs

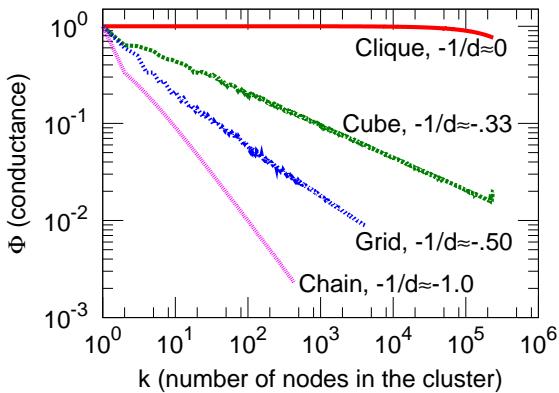
The NCP plot behaves in a characteristic manner for graphs that are “well-embeddable” into an underlying low-dimensional geometric structure. To illustrate this, consider Figure 10.4. In Figure 10.4(a), we show the results for a 1-dimensional chain, a 2-dimensional grid, and a 3-dimensional cube. In each case, the NCP plot is steadily downward sloping as a function of the number of nodes in the smaller cluster. Moreover, the curves are straight lines with a slope equal to $-1/d$, where d is the dimensionality of the underlying grids. In particular, as the underlying dimension increases then the slope of the NCP plot gets less steep. Thus, we observe:

Observation 10.3.2. *If the network under consideration corresponds to a d -dimensional grid, then the NCP plot shows that*

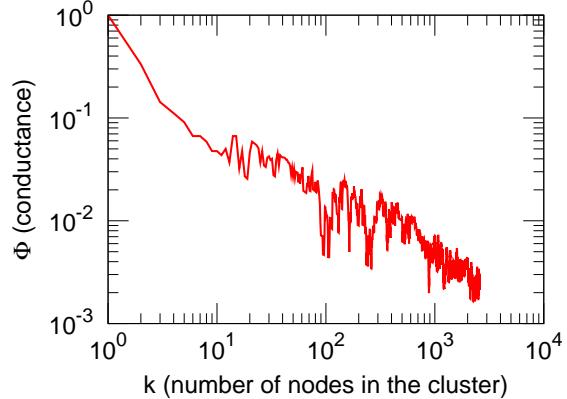
$$-\frac{1}{d} = \frac{\log(\phi(k))}{\log(k)}. \quad (10.9)$$

This is simply a manifestation of the isoperimetric (*i.e.*, surface area to volume) phenomenon: for a grid, the “best” cut is obtained by cutting out a set of adjacent nodes, in which case the surface area (number of edges cut) increases as $O(m^{d-1})$, while the volume (number of vertices/edges inside the cluster) increases as $O(m^d)$.

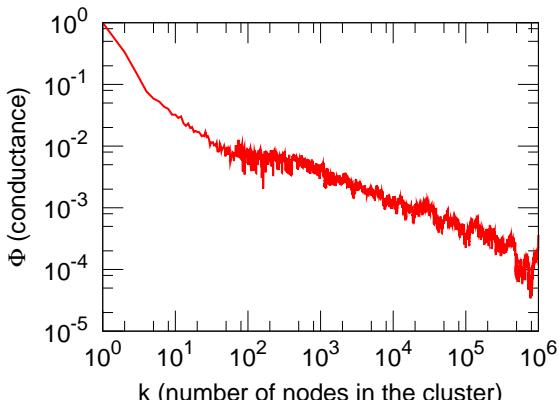
This qualitative phenomenon of a steadily downward sloping NCP plot is quite robust for networks that “live” in a low-dimensional structure, *e.g.*, on a manifold or the surface of the earth. For example, Figure 10.4(b) shows the NCP plot for a power grid network of Western States Power Grid [Watts and Strogatz, 1998], and Figure 10.4(c) shows the NCP plot for a road network of California. These two networks have very different sizes—the power grid network has 4,941 nodes and 6,594 edges, and the road network has



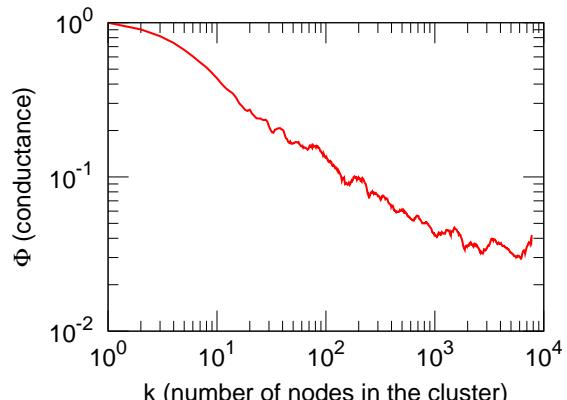
(a) Several low-dimensional meshes.



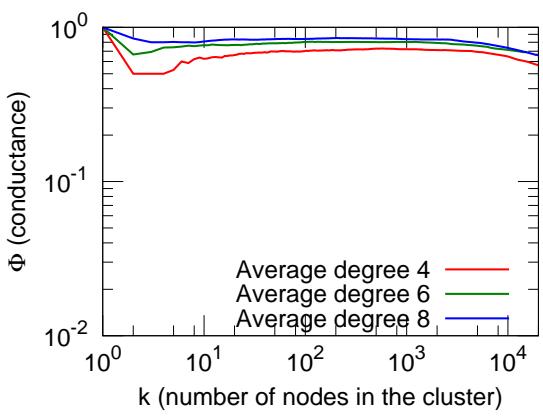
(b) POWERGRID



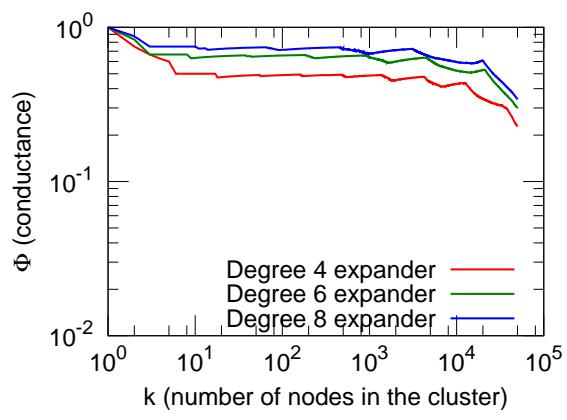
(c) ROAD-CA



(d) Manifold



(e) Expander: dense G_{nm} graph



(f) Expander: union of matchings

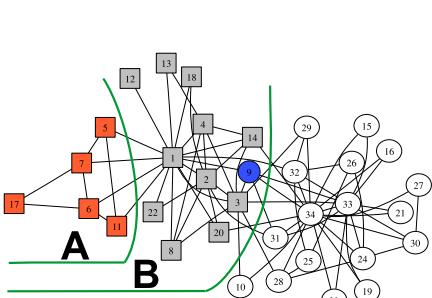
Figure 10.4: Network community profile plots for expander-like graphs and several networks that “live” in low-dimensional spaces. (10.4(a)) A large clique graph, a cube (3d mesh), a grid (2d mesh) and a chain (line). Note that the slope of community profile plot directly corresponds to dimensionality of the graph. (10.4(b)) and (10.4(c)) Two networks on the Earth’s surface and thus that are reasonably well-embeddable in two dimensions. (10.4(d)) A 2d “swiss roll” manifold embedded in 3 dimensions, where every we connected every point to 10 nearest neighbors. (10.4(e)) and (10.4(f)) Two networks that are very good expanders.

1,957,027 nodes and 2,760,388 edges—and they arise in very different application domains. In both cases, however, we see predominantly downward sloping NCP plot, very much similar to the profile of a simple 2-dimensional grid. Indeed, the “best-fit” line for power grid gives the slope of ≈ -0.45 , which by (10.9) suggests that $d \approx 2.2$, which is not far from the “true” dimensionality of 2. Moreover, empirically we observe that minima in the NCP plot correspond to community-like sets, which are occasionally nested. This corresponds to hierarchical community organization. For example, the nodes giving the dip at $k = 19$ are included in the nodes giving the dip at $k = 883$, while dips at $k = 94$ and $k = 105$ are both included in the dip at $k = 262$.

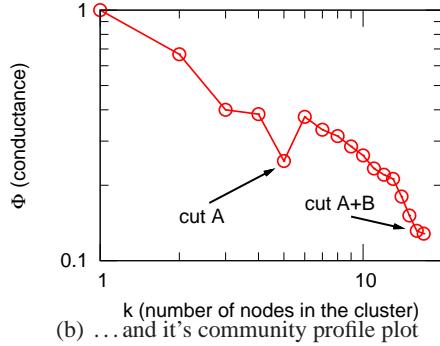
In a similar manner, Figure 10.4(d) shows the profile plot for a graph generated from a “swiss roll” dataset which is commonly examined in the manifold and machine learning literature [Tenenbaum et al., 2000]. In this case, we still observe a downward sloping NCP plot that corresponds to internal dimensionally of the manifold (2 in this case). Finally, Figures 10.4(e) and 10.4(f) show NCP plots for two graphs that are very good expanders. The first is a G_{nm} graph with 100,000 nodes and a number of edges such that the average degree is 4, 6, and 8. The second is a constant degree expander: to make one with degree d , we take the union of d disjoint but otherwise random complete matchings, and we have plotted the results for $d = 4, 6, 8$. In both of these cases, the NCP plot is roughly flat, which we also observed in Figure 10.4(a) for a clique, which is to be expected since the minimum conductance cut in the entire graph cannot be too small for a good expander [Hoory et al., 2006].

Somewhat surprisingly (especially when compared with large networks in Section 10.3.3), a steadily decreasing downward NCP plot is seen for small social networks that have been extensively studied in validating community detection algorithms. Several examples are shown in Figures 10.5. For these networks, the interpretation is similar to that for the low-dimensional networks: the downward slope indicates that as potential communities get larger and larger, there are relatively more intra-edges than inter-edges; and empirically we observe that local minima in the NCP plot correspond to sets of nodes that are plausible communities. Consider, *e.g.*, Zachary’s karate club [Zachary, 1977] network (ZACHARYKARATE), an extensively-analyzed social network [Newman, 2004, 2006b, Karrer et al., 2008]. The network has 34 nodes, each of which represents a member of a karate club, and 78 edges, each of which represent a friendship tie between two members. Figure 10.5(a) depicts the karate club network, and Figure 10.5(b) shows its NCP plot. There are two local minima in the plot: the first dip at $k = 5$ corresponds to the Cut A, and the second dip at $k = 17$ corresponds to Cut B. Note that Cut B, which separates the graph roughly in half, has better conductance value than Cut A. This corresponds with the intuition about the NCP plot derived from studying low-dimensional graphs. Note also that the karate network corresponds well with the intuitive notion of a community, where nodes of the community are densely linked among themselves and there are few edges between nodes of different communities.

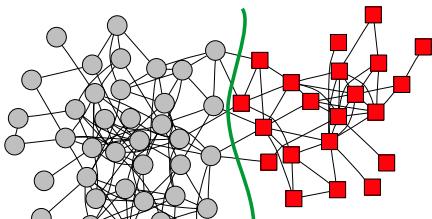
In a similar manner: Figure 10.5(c) shows a social network (with 62 nodes and 159 edges) of interactions within a group of dolphins [Lusseau et al., 2003]; Figure 10.5(e) shows a social network of monks (with 18 nodes representing individual monks and 41 edges representing social ties between pairs of monks) in a cloister [Sampson, 1968]; and Figure 10.5(g) depicts Newman’s network (with 914 collaborations between 379 researchers) of scientists who conduct research on networks [Newman and Girvan, 2004]. For each network, the NCP plot exhibits a downward trend, and it has local minima at cluster sizes that correspond to good communities: the minimum for the dolphins network (Figure 10.5(d)) corresponds to the separation of the network into two communities denoted with different shape and color of the nodes (gray circles versus red squares); the minima of the monk network (Figure 10.5(f)) corresponds to the split of 7 Turks (red squares) and the so-called loyal opposition (gray circles) [Sampson, 1968]; and empirically both local minima and the global minimum in the network science network (Figure 10.5(h)) correspond to



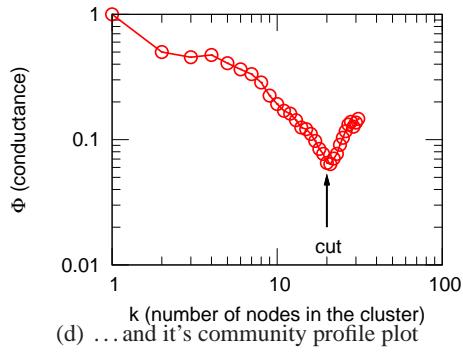
(a) Zachary's karate club network ...



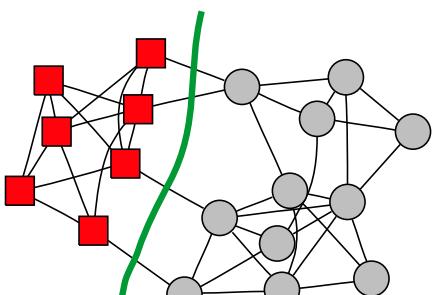
(b) ... and it's community profile plot



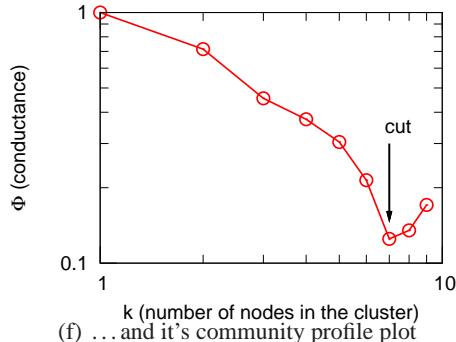
(c) Dolphins social network ...



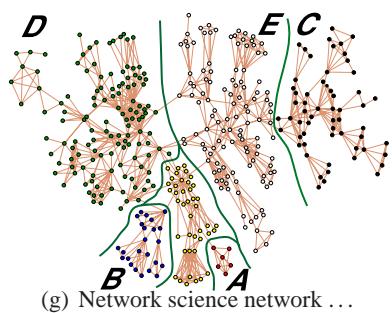
(d) ... and it's community profile plot



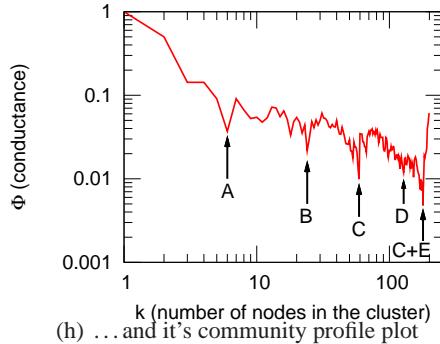
(e) Monks social network ...



(f) ... and it's community profile plot



(g) Network science network ...



(h) ... and it's community profile plot

Figure 10.5: Depiction of several small social networks that are common test sets for community detection algorithms and their network community profile plots. (10.5(a)–10.5(b)) Zachary's karate club network. (10.5(c)–10.5(d)) A network of dolphins. (10.5(e)–10.5(f)) A network of monks. (10.5(g)–10.5(h)) A network of researchers researching networks.

plausible communities. Note that in the last case, the figure also displays hierarchical structure in which case the community defined by Cut C is included in a larger community that has better conductance value.

At this point, we can observe that the following two general observations hold for networks that are well-embeddable in a low-dimensional space and also for small social networks that have been extensively studied and used to validate community detection algorithms. First, minima in the NCP plots, *i.e.*, the best low-conductance cuts of a given size, correspond to communities-like sets of nodes. Second, the NCP plots are generally relatively gradually sloping downwards, meaning that smaller communities can be combined into larger sets of nodes that can also be meaningfully interpreted as communities.

10.3.3 NCP plots for large social and information networks

We have examined NCP plots for each of the networks listed in Tables A.2, A.3 and A.4. In Figure 10.6, we present NCP plots for six of these networks. (These particular networks were chosen to be representative of the wide range of networks we have examined, and for ease of comparison we will compute other properties for them in future sections. See Figures 10.7, 10.8, and 10.9 in Section 10.3.4 for the NCP plots of other networks listed in Tables A.2, A.3 and A.4, and for a discussion of them.) The most striking feature of these plots is that the NCP plot is steadily increasing for nearly its entire range.

Consider, first, the NCP plot for the LIVEJOURNAL01 social network, as shown in Figure 10.6(a), and focus first on the red curve, which presents the results of applying the Local Spectral Algorithm.² We make the following observations:

- Up to a size scale, which empirically is roughly 100 nodes, the slope of the NCP plot is generally sloping downward.
- At that size scale, we observe the global minimum of the NCP plot. This set of nodes as well as others achieving local minima of the NCP plot in the same size range are the “best” communities, according to the conductance measure, in the entire graph.
- These best communities (the best denoted by a square) are barely connected to the rest of the graph, *e.g.*, they are typically connected to the rest of the nodes by a *single* edge.
- Above the size scale of roughly 100 nodes, the NCP plot gradually increases over several orders of magnitude. The “best” communities in the entire graph are quite good (in that they have size roughly 10^2 nodes and conductance scores less than 10^{-3}) whereas the “best” communities of size 10^5 or 10^6 have conductance scores of about 10^{-1} . In between these two size extremes, the conductance scores get gradually worse, although there are numerous local dips and even one relatively large dip between 10^5 and 10^6 nodes.

² The algorithm takes as input two parameters—the seed node and the parameter ϵ that intuitively controls the locality of the computation—and it outputs a set of nodes. For a given seed node and resolution parameter ϵ we obtain a local community profile plot, which tells us about conductance of cuts in vicinity of the seed node. By taking the lower-envelope over community profiles of different seed nodes and ϵ values we obtain the global network community profile plot. For our experiments, we typically considered 100 different values of ϵ . Since very local random walks discover small clusters, in this case we considered every node as a seed node. As we examine larger clusters, the random walk computation spreads farther away from the seed node, in which case the exact choice of seed node becomes less important. Thus, in this case, we sampled fewer seed nodes. Additionally, in our experiments, for each value of ϵ we randomly sampled nodes until each node in the network was visited by random walks starting from, 10 different seed nodes on average.

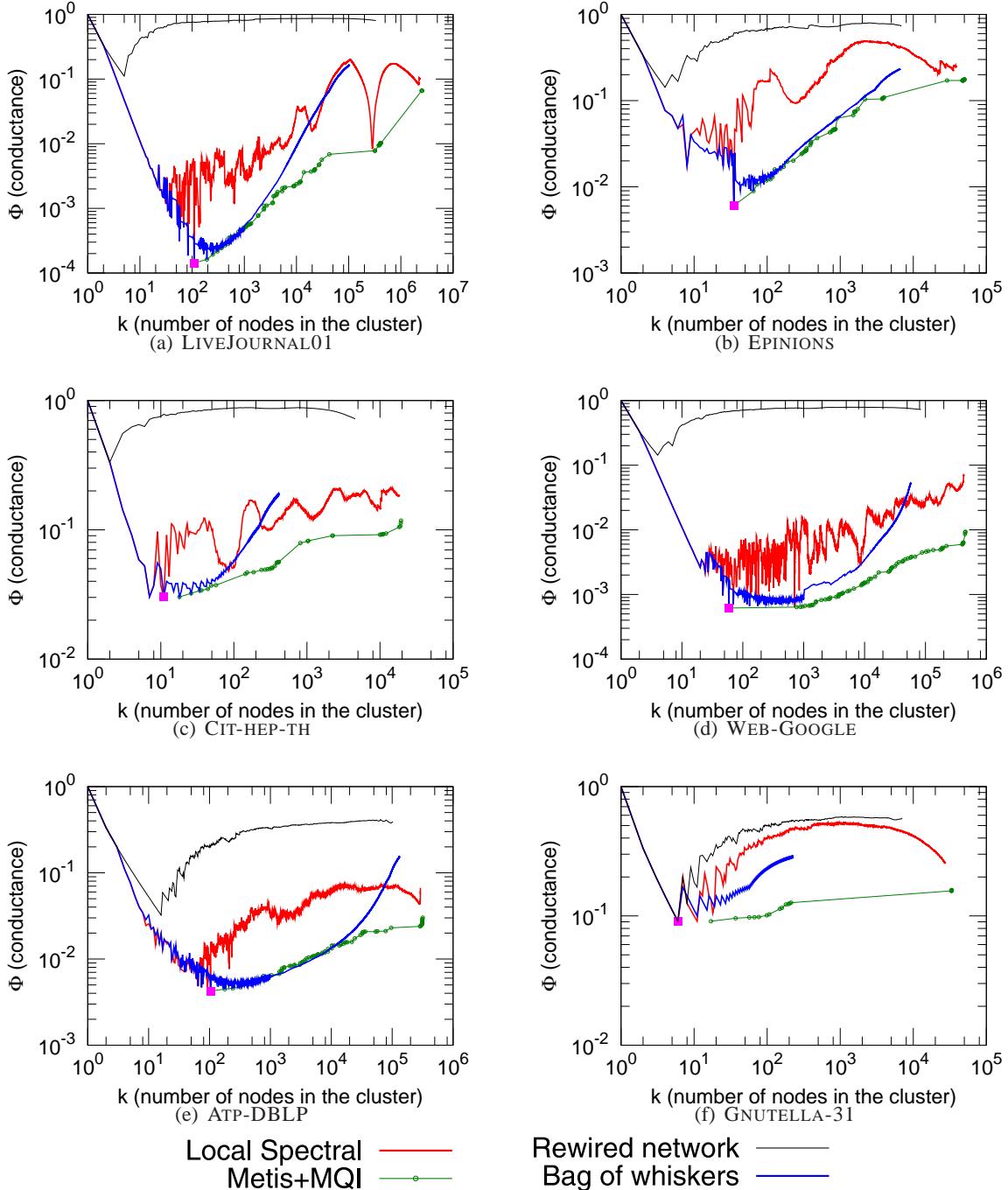


Figure 10.6: Network community profile plots for a representative sample of large networks listed in Tables A.2, A.3 and A.4. The red curves plot the results of the Local Spectral Algorithm on the specified network; green curves plot the results of Metis+MQI; blue curves plot the results of the Bag-of-Whiskers Heuristic; and black curves plot the results of the Local Spectral Algorithm applied to a randomly rewired version of the same network. Notice that in all cases the “best” communities are quite small (typically between 10 and 100 nodes) and that the network community profile plot steadily increases for nearly its entire range. See Figures 10.7, 10.8, and 10.9 for the NCP plots of other networks.

Note that both axes in Figure 10.6 are logarithmic, and thus the upward trend of the NCP plot is over a wide range of size scales. Note also that the green curve plots the results of Metis+MQI (that returns disconnected clusters), and the blue curve plots the results of applying the Bag-of-Whiskers Heuristic, as described in Section 10.4.3. These procedures will be discussed in detail in Sections 10.4 and 10.5.

The black curve in Figure 10.6(a) plots the results of the Local Spectral Algorithm applied to a *rewired version* of the LIVEJOURNAL01 network, *i.e.*, to a random graph conditioned on the same degree distribution as the original network. (We obtain such random graph by starting with the original network and then randomly selecting pairs of edges and rewiring the endpoints. By doing the rewiring long enough, we obtain a random graph that has the same degree sequence as the original network [Milo et al., 2004].)

Interestingly, the NCP of a rewired network first slightly decreases but then increases and flattens out. Several things should be noted:

- The original LIVEJOURNAL01 network has considerably more structure, *i.e.*, deeper/better cuts, than its rewired version, even up to the largest size scales. That is, we observe significantly more structure than would be seen, for example, in a random graph on the same degree sequence.
- Relative to the original network, the “best” community in the rewired graph, *i.e.*, the global minimum of the conductance curve, shifts upward and towards the left. This means that in rewired networks the best conductance clusters get smaller and have worse conductance scores.
- Sets at and near the minimum are small trees that are connected to the core of the random graph by a single edge.
- After the small dip at a very small size scale (≈ 10 nodes), the NCP plot increases to a high level rather quickly. This is due to the absence of structure in the core.

Finally, also note that the variance in the rewired version of the NCP plot (data not shown) is not much larger than the width of the curve in the figure.

We have observed qualitatively similar results in nearly every large social and information network we have examined. For example, several additional examples are presented in Figure 10.6: another network from the class of social networks (EPINIONS, in Figure 10.6(b)); an information/citation network (CIT-HEP-TH, in Figure 10.6(c)); a Web graph (WEB-GOOGLE, in Figure 10.6(d)); a Bipartite affiliation network (ATP-DBLP, in Figure 10.6(e)); and an Internet network (GNUTELLA-31, in Figure 10.6(f)).

Qualitative observations are consistent across the range of network sizes, densities, and different domains from which the networks are drawn. Of course, these six networks are very different than each other—some of these differences are hidden due to the definition of the NCP plot, whereas others are evident. Perhaps the most obvious example of the latter is that even the best cuts in GNUTELLA-31 are not significantly smaller or deeper than in the corresponding rewired network, whereas for WEB-GOOGLE we observe cuts that are orders of magnitude deeper.

Intuitively, the upward trend in the NCP plot means that separating large clusters from the rest of the network is especially expensive. It suggests that larger and larger clusters are “blended in” more and more with the rest of the network. The interpretation we draw, based on these data and data presented in subsequent sections is that, if a density-based concept such as conductance captures our intuitive notion of community goodness and if we model large networks with interaction graphs, then the best possible communities get less and less community-like as they grow in size.

10.3.4 More community profile plots for large social and information networks

Figures 10.7, 10.8, and 10.9 show additional examples of NCP plots for networks from Tables A.2, A.3 and A.4. In the first two rows of Figure 10.7, we have several examples of purely Social networks and two email networks, in the third row we have patent and blog Information/citation networks, and in the final row we have three examples of actor and author Collaboration networks. In Figure 10.8, we see three examples each of Web graphs, Internet networks, Bipartite affiliation networks, and Biological networks. Finally, in the first row of Figure 10.9, we see Low-dimensional networks, including two road and a manifold network; in the second row, we have an IMDB Actor-to-Movie graphs and two subgraphs induced by restricting to individual countries; in the third row, we see three Amazon product co-purchasing networks; and in the final row we see a Yahoo! Answers networks and two subgraphs that are large good conductance cuts from the full network.

For most of these networks, the same four versions of the NCP plot are plotted that were presented in Figure 10.6. Note that, as before, the scale of the vertical axis in these graphs is not all the same; the minima range from 10^{-2} to 10^{-5} . These network datasets are drawn from a wide range of areas, and these graphs contain a wealth of information, a full analysis of which is well beyond the scope of the chapter. Note, however, that the general trends we discussed in Section 10.3.3 still manifest themselves in nearly every network.

The IMDB-RAW07 network is interesting in that its NCP plot does not increase much (at least not the version computed by the Local Spectral Algorithm) and we clearly observe large sets with good conductance values. Upon examination, many of the large good conductance cuts seem to be associated with different language groups. Two things should be noted. First, and not surprisingly, in this network and others, we have observed that there is some sensitivity to how the data are prepared. For example, we obtain somewhat stronger communities if ambiguous nodes (and there are a lot of ambiguous nodes in network datasets with millions of nodes) are removed than if, *e.g.*, they are assigned to a country based on a voting mechanism of some other heuristic. A full analysis of these data preparation issues is beyond the scope of this chapter, but our overall conclusions seem to hold independent of the preparation details. Second, if we examine individual countries—two representative examples are shown—then we see substantially less structure at large size scales.

The Yahoo! Answers social network (see ANSWERS) also has several large cuts with good conductance value—actually, the best cut in the network has more 10^5 nodes. (It is likely that exogenous factors are responsible for these large deep cuts.) Using standard graph partitioning procedures, we obtained four large disjoint clusters consisting of ca. 5,300, 25,400, 27,000, and 290,000 nodes, respectively, corresponding to the four dips (two of which visually overlap) in the NCP plot. We then examined the community profile plots for each of these pieces. The two representative examples of which we show clearly indicate a NCP plot that is much more like other network datasets we have examined.

10.4 More structural observations of our network datasets

We have examined in greater detail our network datasets in order to understand which structural properties are responsible for the observed properties of the NCP plot. We first present statistics for our network datasets in Section 10.4.1. Then, in Section 10.4.2 we describe a heuristic to identify small sets of nodes that have strong connections amongst themselves but that are connected to the remainder of the network by only a single edge. In Section 10.4.3, we show that these “whiskers” (or disjoint unions of them) are

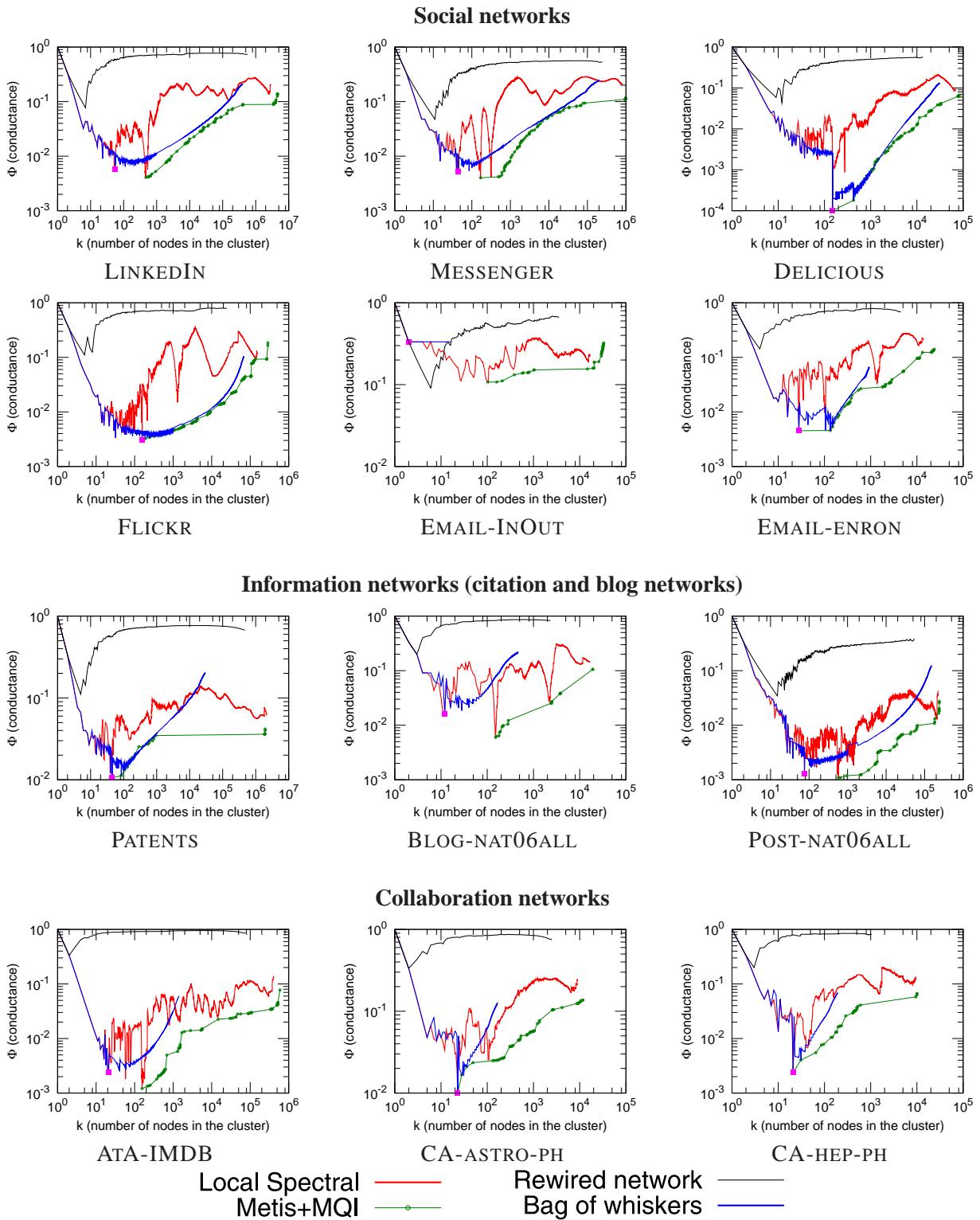


Figure 10.7: Community profile plots of networks from Table A.2.

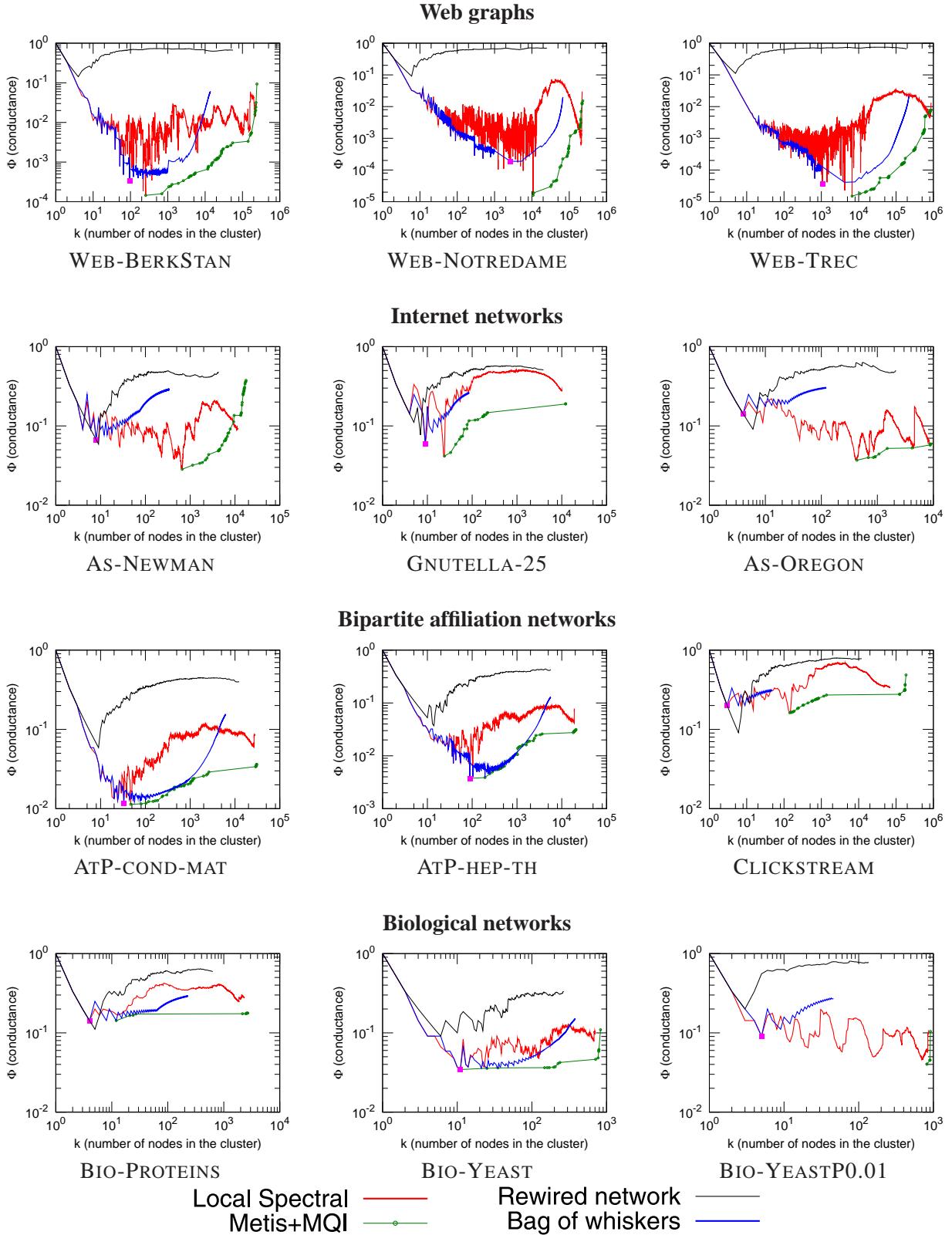


Figure 10.8: Community profile plots of networks from Table A.3.

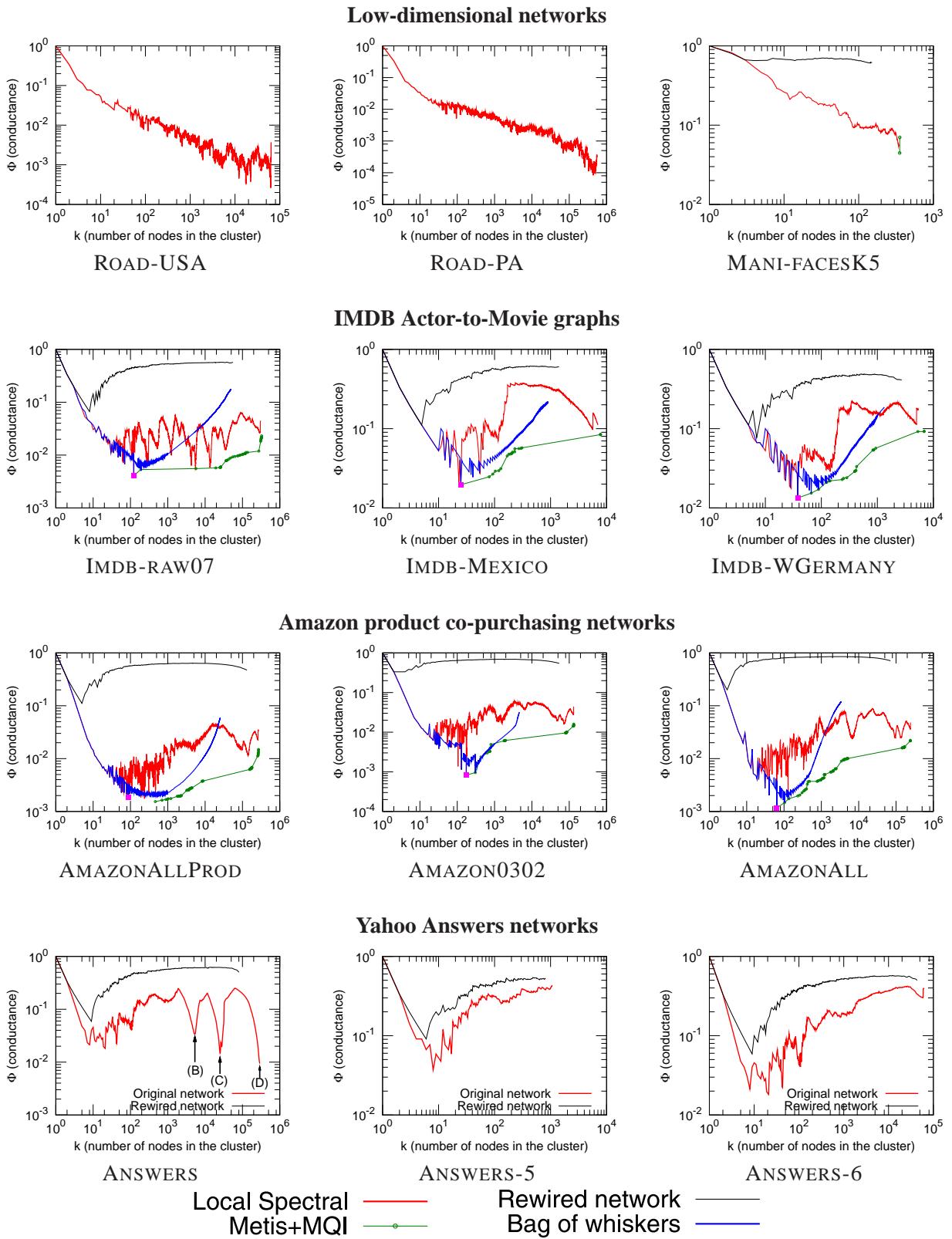


Figure 10.9: Community profile plots of networks from Table A.4, as well as ANSWERS and two sub-pieces of ANSWERS.

often the “best” conductance communities in the network. Last, in Section 10.4.4 we examine NCP plots for networks in which these whiskers have been removed.

10.4.1 General statistics on our network datasets

In Tables A.2, A.3, and A.4, we also present the following statistics for our network datasets: the number of nodes N ; the number of edges E ; the fraction of nodes in the largest biconnected component N_b/N ; the fraction of edges in the largest biconnected component E_b/E ; the average degree $\bar{d} = 2E/N$; the empirical second-order average degree [Chung and Lu, 2006a] \tilde{d} ; average clustering coefficient [Watts and Strogatz, 1998] \bar{C} ; the estimated diameter D ; and the estimated average path length \bar{D} . (The diameter was estimated using the following algorithm: pick a random node, find the farthest node X (via shortest path); move to X and find the farthest node from X ; iterate this procedure until the distance to the farthest node does not increase anymore. The average path length was estimated based on 10,000 randomly sampled nodes.)

In nearly every network we have examined, there is a substantial fraction of nodes that are barely connected to the main part of the network, *i.e.*, that are part of a small cluster of ca. 10 to 100 nodes that are attached to the remainder of the network via one or a small number of edges. In particular, a large fraction of the network is made out of nodes that are not in the biconnected core.³

For example, the EPINIONS network has 75,877 nodes and 405,739 edges, and the core of the network has only 36,111 (47%) nodes and 365,253 (90%) edges. For DELICIOUS, the core is even smaller: it contains only 40% of the nodes, and 65% of the edges. Averaging over our network datasets, we see that the largest biconnected component contains around only 60% of the nodes and 80% of the edges of the original network. This is somewhat akin to the so-called “Jellyfish” model [Tauro et al., 2001, Siganos et al., 2006] (which was proposed as a model for the graph of internet topology) and also to the “Octopus” model (for random power law graphs [Chung and Lu, 2006a], which is described in more detail in Section 10.6.2). Moreover, the global minimum of the NCP plot is nearly always one of these pieces that is connected by only a single edge. Since these small barely-connected pieces seem to have a disproportionately large influence on the community structure of our network datasets, we examine them in greater detail in the next section.

10.4.2 Network “whiskers” and the “core”

We define *whiskers*, or more precisely *1-whiskers*, to be maximal subgraphs that can be detached from the rest of the network by removing a *single* edge. (Occasionally, we use the term whiskers informally to refer to barely connected sets of nodes more generally.) To find 1-whiskers, we employ the following algorithm. Using a depth-first search algorithm, we find the largest biconnected component B of the graph G . (A graph is biconnected if the removal of any single edge does not disconnect the graph.) We then delete all the edges in G that have one of their end points in B . We call the connected components of this new graph G' 1-whiskers, since they correspond to largest subgraphs that can be disconnected from G by removing just a single edge. Recall that Figure 10.2(b) contains a schematic picture a network, including several of its whiskers.

³ In this chapter, we are slightly abusing standard terminology by using the term bi-connectivity to mean 2-edge-connectivity. We are running the classic DFS-based bi-connectivity algorithm, which identifies both bridge edges and articulation nodes, but then we are only knocking out the bridge edges, not the articulation nodes, so we end up with 2-edge-connected pieces.

There is a wide range of whisker sizes and shapes. Figure 10.10 shows the distribution of 1-whisker sizes for a representative selection of our network datasets. Empirically, 1-whisker size distribution is heavy-tailed, with the largest whisker size ranging from around less than 10 to well above 100. The largest whiskers in co-authorship and citation networks have around 10 nodes, whiskers in bipartite graphs also tend to be small, and very large whiskers are found in a web graph. Figure 10.10 also compares the size of the whiskers with the sizes of whiskers in a rewired version of the same network. (The first thing to note is that due to the sparsity of the networks, the rewired versions all have whiskers.) In rewired networks the whiskers tend to be much smaller than in the original network. A particularly noteworthy exception is found in the Autonomous systems networks and the GNUTELLA-31 network. (See Figure 10.10(f) for an example of the latter.) In these cases, the whiskers are so small that even the rewired version of the network has more and larger whiskers. This makes sense, given how those networks were designed: clearly, many large whiskers would have negative effects on the Internet connectivity in case of link failures.

Figure 10.11 shows the ten largest whiskers of the EPINIONS social network, the full size distribution of which was plotted in Figure 10.10(b), and Figure 10.12 shows the ten largest whiskers of the CA-COND-MAT co-authorship network. In these networks, the whiskers have on the order of 10 nodes, and they are seen to have a rich internal structure. Similar but substantially more complex figures could be generated for networks with larger whiskers. In general, the results we observe are consistent with a knowledge of the fields from which the particular datasets have been drawn. For example, in WEB-GOOGLE we see very large whiskers. This probably represents a well-connected network between the main categories of a website (*e.g.*, different projects), while the individual project websites have a main index page that then points to the rest of the documents.

The discrepancy between the sizes of the whiskers in the original and the rewired networks gives hints that real networks have much richer structure than that imposed by their heavy-tailed degree distribution. One might ask whether the conclusion from this is that real-world graphs should be thought of as being somewhat like sparse random graphs, since, *e.g.*, both have whiskers, or should be thought of as very different than sparse random graphs, since, *e.g.*, the whiskers have much more internal structure. We will return to this issue in Section 10.6.

10.4.3 Bags of whiskers and communities of composed whiskers

Empirically, if one looks at the sets of nodes achieving the minimum in the NCP plot (green Metis+MQI curve), then before the global NCP minimum communities are whiskers and above that size scale they are often unions of disjoint whiskers. To understand the extent to which these whiskers and unions of whiskers are responsible for the “best” conductance sets of different sizes, we have developed the *Bag-of-Whiskers Heuristic*. We artificially compose “communities” from disconnected whiskers and measure conductance of such clusters. Clearly, interpreting and relating such communities to real-world communities makes little sense as these communities are in fact disconnected.

In more detail, we performed the following experiment: suppose we have a set $W = \{w_1, w_2, \dots\}$ of whiskers. In order to construct the optimal conductance cluster of size k , we need to solve the following problem: find a set C of whiskers such that $\sum_{i \in C} N(w_i) = k$ and $\sum_{i \in C} \frac{d(w_i)}{|C|}$ is maximized, where $N(w_i)$ is the number of nodes in w_i and $d(w_i)$ is its total internal degree. We then use a dynamic programming to get an approximate solution to this problem. This way, for each size k , we find a cluster that is composed solely from (disconnected) whiskers. Figure 10.6 as well as Figures 10.7, 10.8 and 10.9 show the results of this heuristic applied to many of our network datasets (blue curve).

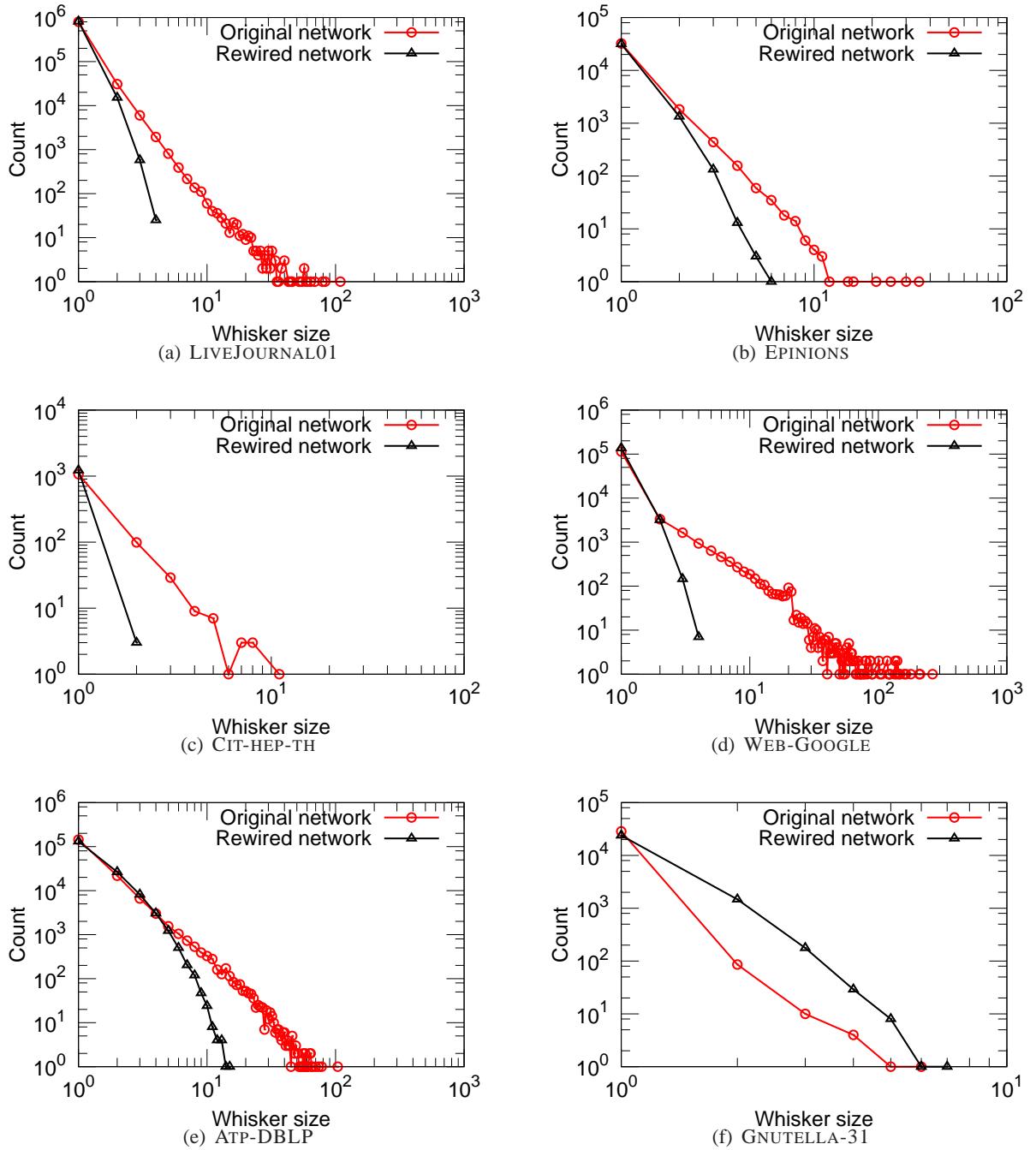


Figure 10.10: Distribution of whisker sizes in the true network and the rewired network (random graph with same degree distribution) for the six networks presented in Figure 10.6. The ten largest whiskers for the EPINIONS social network (the full distribution of which is presented here in panel (b)) are presented in Figure 10.11.

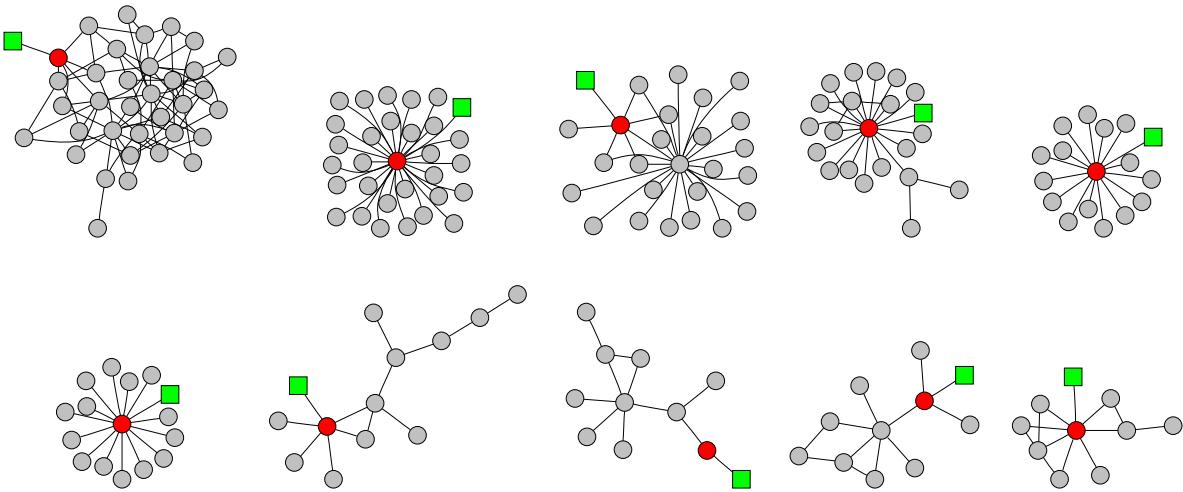


Figure 10.11: Ten largest whiskers of the EPINIONS social network. The green square node is the node from the bi-connected core of the network to which the whisker is connected. For visual clarity, the whisker node that connects to the core of the network is displayed in red, and thus it is the edge between the red circle and the green square node that if cut disconnects the whisker from the core. The distribution of whisker sizes and comparison to rewired network is plotted in Figure 10.10(b).

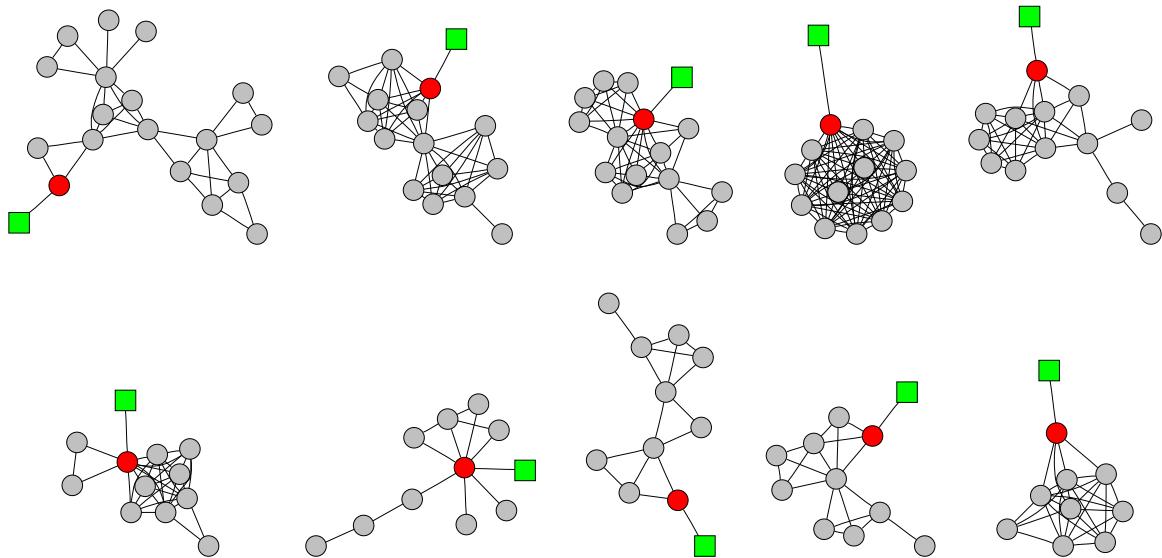


Figure 10.12: Ten largest whiskers of the CA-COND-MAT co-authorship network. The green square node belongs to the network core, and by cutting the edge connecting it with red circular node we separate the community of circles from the rest of the network (depicted as a green square).

There are several observations we can make:

- The largest whisker (denoted with a red square) is the lowest point in nearly all NCP plots. This means that the best conductance community is in a sense trivial as it cuts just a single edge, and in addition that a very simple heuristic can find this set.
- For community size below the critical size of ≈ 100 nodes (*i.e.*, of size smaller than the largest whisker), the best community in the network is actually a whisker and can be cut by a single edge (blue and red curve overlap).
- For community size larger than the critical size of ≈ 100 , the Bag-of-Whiskers communities have better scores than the internally well-connected communities extracted by Local Spectral (red curve). The shape of this blue curve in that size region depends on the distribution of sizes of whiskers, but in nearly every case it is seen to yield better conductance sets than the Local Spectral Algorithm.

Moreover, the Bag-of-Whiskers Heuristic often almost exactly agrees with results from Metis+MQI (green curve). In particular, the best conductance sets of a given size are often disconnected, and when they are connected they are often only tenuously connected. Thus, if one only cares about finding good cuts then the best cuts in these large sparse graphs are obtained by composing unrelated disconnected pieces. Intuitively, a compact cluster is internally well and evenly connected. Possible measures for cluster compactness include: cluster connectedness, diameter, conductance of the cut inside the cluster, ratio of conductance of the cut outside versus the cut inside. We discuss this in more detail in Section 10.5.

10.4.4 NCP of networks with no 1-whiskers

Given the surprisingly significant effect on the community structure of real-world networks that whiskers and unions of disjoint whiskers have, one might wonder whether we see something qualitatively different if we consider a real-world network in which these barely-connected pieces have been removed. To study this, we found all 1-whiskers and removed them from our networks, using the procedure we described in Section 10.4.2, *i.e.*, we selected the largest biconnected component for each of our network datasets. This way, we kept only the network core, and we then computed the NCP plots for these modified networks. Figure 10.13 shows the NCP plots of networks constructed when we remove whiskers (*i.e.*, keep only the network core) for the six networks we studied in detail before.

Notice that whisker removal does not change the NCP plot much: the plot shifts slightly upward, but the general trends remain the same. Upon examination, the global minimum occurs with a “2-whisker” that is connected by two edges to the remainder of the graph. Intuitively, the largest biconnected core has a large number of barely connected pieces—connected now by two edges rather than by one edge—and thus the “core” itself has a core-periphery structure. Since the “volume” for these pieces is similar to that for the original whiskers, whereas the “surface area” is a factor of two larger, the conductance value is roughly a factor of two worse. Thus, although we have been discussing 1-whiskers in this section, one should really view them as the simplest example of weakly-connected pieces that exert a significant effect on the community structure in large real-world networks.

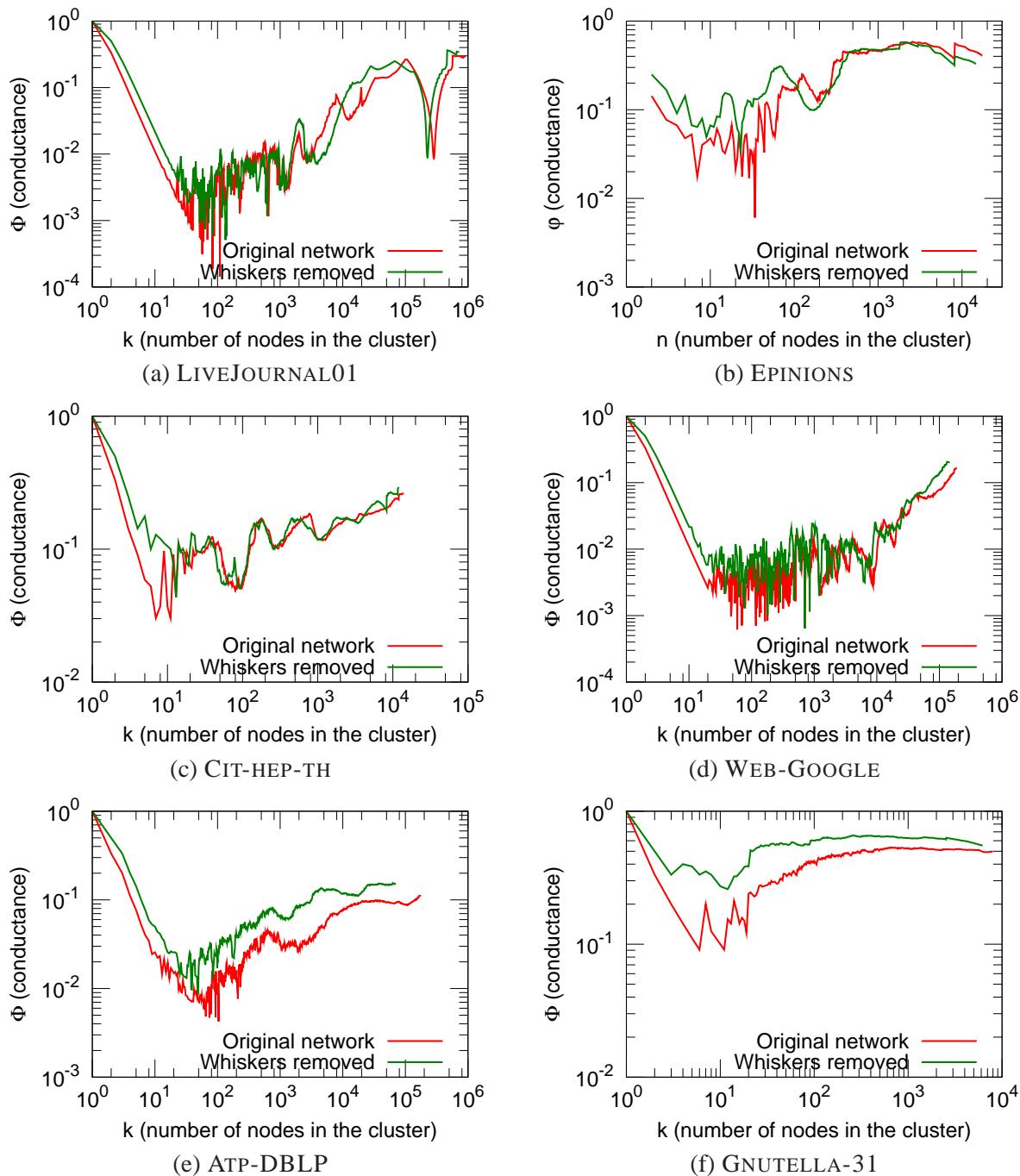


Figure 10.13: Network community profile plots with (in red) and without (in green) 1-whiskers, for each of the six networks shown Figure 10.6. Whiskers were removed as described in the text. In the former case, we plot results for the full network, and in the latter case, we plot results for the largest bi-connected component.

10.5 Comparison to other algorithms

So far, we have been primarily relying on two graph partitioning algorithms: a Local Spectral Algorithm and Metis+MQI. Next, we want to demonstrate that what we are observing is a true structural property of our network datasets, rather than properties of our algorithms; and we want to use the differences between different approximation algorithms to further highlight structural properties of our network datasets. In this section we discuss several meta-issues related to this, including whether or not our algorithms are sufficiently powerful to recover the true shape of the minimal conductance curves, and whether we should actually be trying to optimize a slightly different measure that combines conductance of the separating cut with the piece compactness.

Recall that we defined the NCP plot to be a curve showing the minimum conductance ϕ as a function of piece size k . Finding the points on this curve is NP-hard. Any cut that we find will only provide an upper bound on the true minimum at the resulting piece's size. Given that fact, how confident can we be that the curve of upper bounds that we have computed has the same rising or falling shape as the true curve?

One method for finding out whether any given algorithm is doing a good job of pushing down the upper bounding curve in a non-size-biased way is to compare its curves for numerous graphs with those produced by other algorithms. In such experiments, it is good if the algorithms are very powerful and also independent of each other. We have done extensive experiments along these lines, and our choice of Local Spectral and Metis+MQI as the two algorithms for the main body of this chapter was based on the results. In Section 10.5.1 we mention a few interesting points related to this.

A different method for reducing our uncertainty about the shape of the true curve would be to also compute lower bounds on the curve. Ideally, one would compute a complete curve of tight lower bounds, leaving a thin band between the upper- and lower-bounding curves, which would make the rising or falling shape of the true curve obvious. In Section 10.5.2 we discuss some experiments with lower bounds. Although we only obtained a few lower bounds rather than a full curve, the results are consistent with our main results obtained from upper-bounding curves.

Finally, in Section 10.5.3 we will discuss our decision to use the Local Spectral algorithm in addition to Metis+MQI in the main body of the chapter, despite the fact that Metis+MQI clearly dominates Local Spectral at the nominal task of finding the lowest possible upper bounding curve for the minimal conductance curve. The reason for this decision is that Local Spectral often returns “nicer” and more “compact” pieces because rather than minimizing conductance alone, it optimizes a slightly different measure that produces a compromise between the conductance of the bounding cut and the “compactness” of the resulting piece.

10.5.1 Cross-checking between algorithms

As just mentioned, one way to gain some confidence in the upper bounding curves produced by a given algorithm is to compare them with the curves produced by other algorithms that are as strong as possible, and as independent as possible. We have extensively experimented with several variants of the global spectral method, both the usual eigenvector-based embedding on a line, and an SDP-based embedding on a hypersphere, both with the usual hyperplane-sweep rounding method and a fancier flow-based rounding method which includes MQI as the last step. In addition, special post-processing can be done to obtain either connected or disconnected sets. After examining the output of those 8 comparatively expensive al-

gorithms on more than 100 graphs, we found that our two cheaper main algorithms did miss an occasional cut on an occasional graph, but nothing at all serious enough to change our main conclusions. All of those detailed results are suppressed in this chapter.

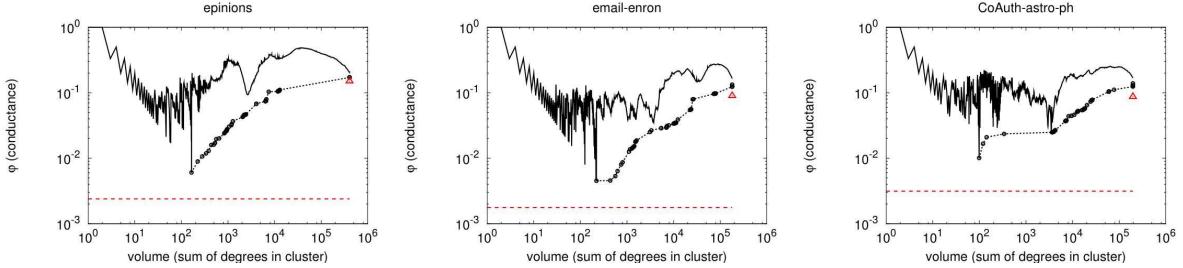
We have also done experiments with a practical version of the Leighton-Rao algorithm [Leighton and Rao, 1988, 1999], similar to the implementation described in [Lang and Rao, 1993] and [Lang and Rao, 2004]. These results are especially interesting because the Leighton-Rao algorithm, which is based on multi-commodity flow, provides a completely independent check on Metis, and on Spectral Methods generally, and therefore on our two main algorithms, namely Metis+MQI and Local Spectral. The Leighton-Rao algorithm has two phases. In the first phase, edge congestions are produced by routing a large number of commodities through the network. We adapted our program to optimize conductance (rather than ordinary ratio cut score) by letting the expected demand between a pair of nodes be proportional to the product of their degrees. In the second phase, a rounding algorithm is used to convert edge congestions into actual cuts. Our method was to sweep over node orderings produced by running Prim's MST algorithm on the congestion graph, starting from a large number of different initial nodes, using a range of different scales to avoid quadratic run time. We used two variations of this method, one that only produces connected sets, and another one that can also produce disconnected sets.

In the second row of Figure 10.14, we show Leighton-Rao curves for three example graphs. Our standard Local Spectral and Metis+MQI curves are drawn in black, while the Leighton-Rao curves for connected and possibly disconnected sets are drawn in green and magenta respectively. We note that for small to medium scales, the Leighton-Rao curves for connected sets resemble the Local Spectral curves, while the Leighton-Rao curves for possibly disconnected sets resemble the Metis+MQI curves. This is big hint about the structure of the sets produced by Local Spectral and Metis+MQI, that we will discuss further in Section 10.5.3.

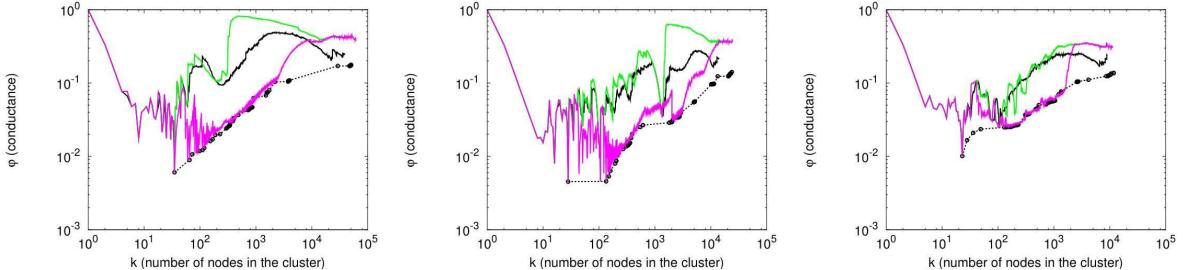
At large scales, the Leighton-Rao curves for these example graphs shoot up and become much worse than our standard curves. This is not surprising because expander graphs are known to be the worst case input for the Leighton-Rao approximation guarantee, and we believe that these graphs contain an expander-like core that is necessarily encountered at large scales. We remark that Leighton-Rao does not work poorly at large scales on every kind of graph. (In fact, for large low-dimensional mesh-like graphs, Leighton-Rao is a very cheap and effective method for finding cuts at all scales, while our local spectral method becomes impractically slow at medium to large scales. We will not discuss this point further, except to note that in the main body of the chapter we have silently substituted Leighton-Rao curves for local spectral curves for the large road networks and similar graphs.)

We have now covered the main theoretical algorithms that are practical enough to actually run, which are based on spectral embeddings and on multi-commodity flow. Starting with [Arora et al., 2004b], there has been a recent burst of theoretical activity showing that spectral and flow-based ideas, which were already known to have complementary strengths and weaknesses, can in fact be combined to obtain the best ever approximations. At present none of the resulting algorithms are sufficiently practical at the sizes that we require, so they were not included in this study.

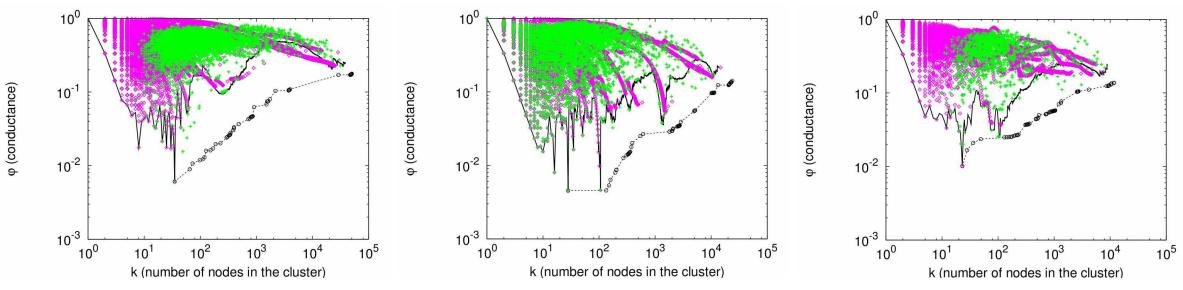
Finally, we mention that in addition to the above theoretically-based practical methods for finding low-conductance cuts, there exist a very large number of heuristic graph clustering methods. We have tried a number of them, including Graclus [Dhillon et al., 2007] and Newman's modularity optimizing program (we refer to it as Dendrogram) [Girvan and Newman, 2002]. Graclus attempts to find a partitioning of a graph into pieces bounded by low-conductance cuts using a kernel k-means algorithm. We ran Graclus repeatedly, asking for $2, 3, \dots, i, \dots, i * \sqrt{2}, \dots$ pieces. Then we measured the size and conductance of



Lower bounds on the conductance of best cut in the network.



Leighton-Rao: connected clusters (green), disconnected clusters (magenta).



NCP plots obtained by Graclus and Newman's Dendrogram algorithm.

— LocSpec --- MetMQI	△ SDP LB - - - Spec LB	— Conn LRao — Disconn LR	+ Graclus ◊ Dendro
---	---	--	---

Figure 10.14: Result of other algorithms for three networks: EPINIONS, EMAIL-ENRON, and CA-ASTRO-PH. Top row plots (in black) conductance curves as obtained by Local Spectral and Metis+MQI. Top row also shows lower bounds on conductance of any cut (Spectral lower bound, dashed line) and the cut separating the graph in half (SDP lower bound, red triangle). Middle row shows NCP plots for connected (green) and disconnected (magenta) pieces from our implementation of the Leighton-Rao algorithm. Bottom row shows the conductance of some cuts found by Graclus and by Newman's Dendrogram algorithm. The overall conclusion is that the qualitative shape of the NCP plots is a structural property of large networks and the plot remains practically unchanged regardless of what particular community detection algorithm we use.

all of the resulting pieces. Newman's Dendrogram program constructs a recursive partitioning of a graph (that is, a dendrogram) from the bottom up by repeatedly deleting the surviving edge with the highest betweenness centrality. A flat partitioning could then be obtained by cutting at the level which gives the highest modularity score, but instead of doing that, we measured the size of conductance of every piece defined by a subtree in the dendrogram.

In the bottom row of Figure 10.14, we present these results as scatterplots. Again our two standard curves are drawn in black. No Graclus or Dendrogram point lies below the Metis+MQI curve. The lower-envelopes of the points are roughly similar to those produced by Local Spectral.

Our main point with these experiments is that the lowest points produced by either Graclus or Dendrogram gradually rise as one moves from small scales to larger scales, so in principle we could have made the same observations about the structure of large social and information networks by running one of those easily downloadable programs instead of the algorithms that we did run. We chose the algorithms we did due to their speed and power, although they may not be as familiar to many readers.

10.5.2 Lower bounds on cut conductance

As mentioned above, our main arguments are all based on curves which are actually upper bounds on the true minimum conductance curve. To get a better idea of how good those upper bounds are, we also compute some lower bounds. Here we will discuss the spectral lower bound [Chung, 1997] on the conductance of cuts of arbitrary balance, and we will also discuss a related SDP-based lower bound [Burer and Monteiro, 2003] on the conductance of any cut that divides the graph into two pieces of equal volume.

First, we introduce the following notation: \vec{d} is a column vector of the graph's node degrees; D is a square matrix whose only nonzero entries are the graph's node degrees on the diagonal; A is the adjacency matrix of G ; $L = D - A$ is then the non-normalized Laplacian matrix of G ; $\mathbf{1}$ is vector of 1's; and $A \bullet B = \text{trace}(A^T B)$ is the matrix dot-product operator.

Now, consider the following optimization problem (which is well known to be equivalent to an eigenproblem):

$$\lambda_G = \min \left\{ \frac{x^T L x}{x^T D x} : x \perp \vec{d}, x \neq 0 \right\}.$$

Let \hat{x} be a vector achieving the minimum value λ_G . Then $\frac{\lambda_G}{2}$ is the spectral lower bound on the conductance of any cut in the graph, regardless of balance, while \hat{x} defines a spectral embedding of the graph on a line, to which rounding algorithms can be applied to obtain actual cuts that can serve as upper bounds at various sizes.

Next, we discuss an SDP-based lower bound on cuts which partition the graph into two sets of exactly equal volume. Consider:

$$\mathcal{C}_G = \min \left\{ \frac{1}{4} L \bullet Y : \text{diag}(Y) = \mathbf{1}, Y \bullet (\vec{d} \vec{d}^T) = 0, Y \succeq 0 \right\},$$

and let \hat{Y} be a matrix achieving the minimum value \mathcal{C}_G . Then \mathcal{C}_G is a lower bound on the weight of any cut with perfect volume balance, and $2\mathcal{C}_G/\text{Vol}(G)$ is a lower bound on the conductance of any cut with perfect volume balance. We briefly mention that since $Y \succeq 0$, we can view Y as a Gram matrix that can be factored as RR^T . Then the rows of R are the coordinates of an embedding of the graph on a hypersphere. Again, rounding algorithms can be applied to the embedding to obtain actual cuts that can serve as upper bounds.

The spectral and SDP embeddings defined here were the basis for the extensive experiments with global spectral partitioning methods that were alluded to in Section 10.5.1. However, in this section, it is the

lower bounds that concern us. In the top row of Figure 10.14, we present the spectral and SDP lower bounds for three example graphs. The spectral lower bound, which applies to cuts of any balance, is drawn as a horizontal line which appears near the bottom of each plot. The SDP lower bound, which only applies to cuts separating a specific volume, namely $\text{Vol}(G)/2$, appears as an upwards-pointing triangle near the right side of the each plot. (Note that plotting this point required us to use volume rather than number of nodes for the x-axis of these three plots.)

Clearly, for these graphs, the lower bound at $\text{Vol}(G)/2$, is higher than the spectral lower bound which applies at smaller scales. More importantly, the lower bound at $\text{Vol}(G)/2$, is higher than our *upper* bounds at many smaller scales, so the true curve must go up, at least at the very end, as one moves from small to large scales.

Take, for example, the top left plot of Figure 10.14 where in black we plot the conductance curves obtained by our (Local Spectral and Metis+MQI) algorithms. With a red dashed line we also plot the lower bound of best possible cut in the network, and with red triangle we plot the lower bound for the cut that separates the graph in two equal volume parts. Thus, the true conductance curve (which is intractable to compute) lies below black but above red line and red triangle. This also demonstrates that the conductance curve which starts at upper left corner of the NCP plot first goes down and reaches the minimum close to the horizontal dashed line (Spectral lower bound) and then sharply rise and ends up above the red triangle (SDP lower bound). This verifies that our conductance curves and obtained NCP plots are not the artifacts of community detection algorithms we employed.

Finally, in Table 10.1 we list for about 40 graphs the spectral and SDP lower bounds on overall conductance and on volume-bisecting conductance, and also the ratio between the two. It is interesting to see that for these graphs this ratio of lower bounds does a fairly good job of discriminating between falling-NCP-plot graphs, which have a small ratio, and rising-NCP-plot graphs, which have a large ratio. Small networks (like COLLEGEFOOTBALL, ZACHARYKARATE and MONKSNETWORK) have downward NCP plot and a small ratio of the SDP and Spectral lower bounds. On the other hand large networks (*e.g.*, EPINIONS or ANSWERS-3) that have downward and then upward NCP plot (as in Figure 10.2(a)) have large ratio of the two lower bounds. This is further evidence that small networks have fundamentally different community structure from large networks and that one has to examine very large networks to observe the gradual absence of communities of size above ≈ 100 nodes.

10.5.3 Local Spectral and Metis+MQI

In this section we discuss our rationale for using Local Spectral in addition to Metis+MQI as one of our two main algorithms for finding sets bounded by low conductance cuts. This choice requires some justification because the NCP plots are intended to show the tightest possible upper bound on the lowest conductance cut for each piece size, while the curve for Local Spectral is generally above that for Metis+MQI.

Our reason for using Local Spectral in addition to Metis+MQI is that Local Spectral returns pieces that are internally “nicer”. For graphs with a rising NCP plot, we have found that many of the low conductance sets returned by Metis+MQI (or Leighton-Rao, or the Bag-of-Whiskers Heuristic) are actually *disconnected*. Since internally disconnected sets are not very satisfying “communities”, it is natural to wonder about NCP plot-style curves with the additional requirement that pieces must be internally well connected. In Section 10.5.1, we generated such a curve using Leighton-Rao, and found that the curve corresponding to connected pieces was higher than a curve allowing disconnected sets.

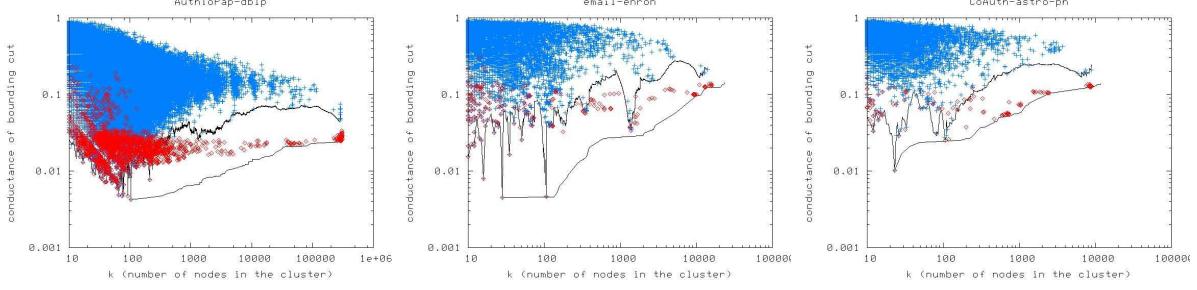
Network	Spectral lowerbnd on ϕ , any size.	SDP lowerbnd on ϕ , at $\text{Vol}(G)/2$	ratio of lower bnds	Network	Spectral lowerbnd on ϕ , any size.	SDP lowerbnd on ϕ , at $\text{Vol}(G)/2$	ratio of lower bnds
COLLEGEFOOTBALL	0.068402	0.091017	1.330624	GNUTELLA-25	0.014185	0.131032	9.237332
MONKSNETWORK	0.069660	0.117117	1.681269	ANSWERS-2	0.009660	0.107422	11.120081
ZACHARYKARATE	0.066136	0.127625	1.929736	CA-COND-MAT	0.003593	0.047064	13.098027
POWERGRID	0.000136	0.000268	1.978484	ANSWERS-1	0.011896	0.159251	13.386528
POLITICALBOOKS	0.018902	0.038031	2.011991	IMDB-FRANCE	0.003462	0.048010	13.867591
POLITICALBLOGS	0.040720	0.084052	2.064157	ANSWERS-5	0.008714	0.124703	14.311255
RB-HIERARCHICAL	0.011930	0.030335	2.542792	IMDB-MEXICO	0.003893	0.070345	18.067513
EMAIL-INOUT	0.038669	0.113367	2.931752	CA-GR-QC	0.000934	0.017421	18.659710
NETWORKSCIENCE	0.001513	0.004502	2.974695	ATP-HEP-TH	0.000514	0.009714	18.899660
AS-OREGON	0.012543	0.042976	3.426417	ATP-HEP-PH	0.000723	0.013770	19.040287
BLOG-NAT05-6M	0.031604	0.108979	3.448250	IMDB-WGERMANY	0.003025	0.065158	21.538867
IMDB-INDIA	0.009104	0.033318	3.659573	ATP-ASTRO-PH	0.001183	0.027256	23.036835
CIT-HEP-PH	0.007858	0.029243	3.721553	CA-HEP-TH	0.001561	0.041125	26.350412
BIO-PROTEINS	0.033714	0.126137	3.741358	CA-ASTRO-PH	0.003143	0.086890	27.648094
AS-ROUTEVIEWS	0.018681	0.070462	3.771821	IMDB-UK	0.001283	0.036572	28.514376
GNUTELLA-31	0.029946	0.118711	3.964127	IMDB-GERMANY	0.000661	0.021017	31.810460
IMDB-JAPAN	0.003327	0.013396	4.026721	BLOG-NAT06ALL	0.002361	0.092908	39.350874
GNUTELLA-30	0.030621	0.124929	4.079853	IMDB-ITALY	0.000679	0.031954	47.077242
DOLPHINSNETWORK	0.019762	0.103676	5.246171	EMAIL-ENRON	0.001763	0.089876	50.965424
AS-NEWMAN	0.009681	0.058952	6.089191	CA-HEP-PH	0.000889	0.052249	58.755927
ATP-GR-QC	0.000846	0.006040	7.141270	EPINIONS	0.002395	0.150242	62.739252
CIT-HEP-TH	0.009193	0.068880	7.492522	ANSWERS-3	0.002636	0.185340	70.306807
ATP-COND-MAT	0.001703	0.013452	7.897650	IMDB-SPAIN	0.000562	0.046327	82.397702

Table 10.1: Lower bounds on the conductance for our network datasets. Recall that the spectral lower bound applies to any cut, while the SDP lower bound applies to cuts at a specified volume fraction, taken here to be half. See the top row of Figure 10.14 for plots for three of these networks.

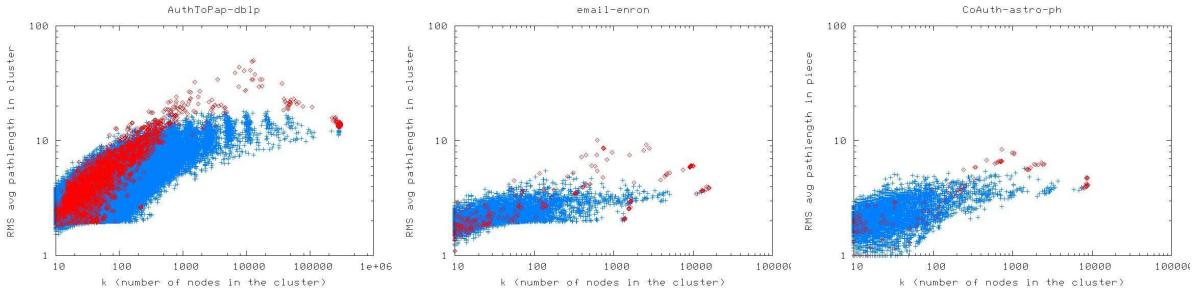
In the top row of Figure 10.15, we show scatter plots illustrating a similar comparison between the conductance of the cuts bounding connected pieces generated by Local Spectral and by Metis+MQI. Our method for getting connected pieces from Metis+MQI here is simply to separately measure each of the pieces in a disconnected set. The blue points in the figures show the conductance of some cuts found by Local Spectral. The red points show the conductance of some cuts found by Metis+MQI. Apparently, Local Spectral and Metis+MQI find similar pieces at very small scales, but at slightly larger scales a gap opens up between the red cloud and the blue cloud. In other words, at those scales Metis+MQI is finding lower conductance cuts than Local Spectral, even when the pieces must be internally connected.

However, there is still a measurable sense in which the Local Spectral pieces are “nicer” and more “compact,” as shown in the second row of scatter plots in Figure 10.15. For each of the same pieces for which we plotted a conductance in the top row, we are now plotting the average shortest path length between random node pairs in that piece. In these plots, we see that in the same size range where Metis+MQI is generating clearly lower conductance connected sets, we now see that Local Spectral is generating pieces with clearly shorter internal paths. In other words, the Local Spectral pieces are more “compact”.

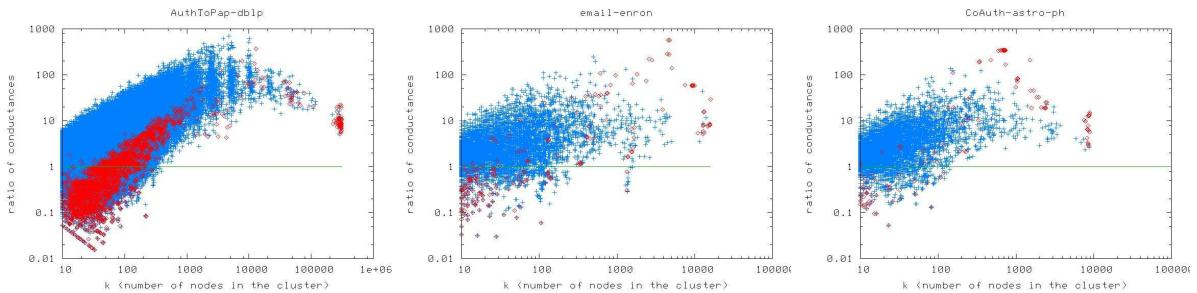
Last, in Figure 10.16, we further illustrate this point with drawings of some example subgraphs. The two subgraphs shown on the left of Figure 10.16 were found by Local Spectral, while the two subgraphs shown on the right of Figure 10.16 were found by Metis+MQI. Clearly, these two pairs of subgraphs have a



Conductance of connected clusters found by Local Spectral (blue) and Metis+MQI (red)



Cluster compactness: average shortest path length



Cluster compactness: external vs. internal conductance

Figure 10.15: Result of comparing Local Spectral (blue) and Metis+MQI (red) on connected clusters for three networks: ATP-DBLP, EMAIL-ENRON, and CA-ASTRO-PH. In the top row, we plot the conductance of the bounding cut. In the middle row, we plot the average shortest path length in the cluster. In the bottom row, we plot the ratio of the external conductance to the internal conductance. Observe that generally Metis+MQI yields better (lower conductance) cuts while Local Spectral yields pieces that are more compact: they have shorter path lengths and internal connectivity.

qualitatively different appearance, with the Metis+MQI pieces looking longer and stringier than the Local Spectral pieces. All of these subgraphs contain roughly 500 nodes, which is a bit more than the natural cluster size for that graph, and thus the differences between the algorithms start to show up. In these cases, Local Spectral has grown a cluster out a bit past its natural boundaries (thus the spokes), while Metis+MQI has strung together a couple of different sparsely connected clusters. (We remark that the tendency of Local Spectral to trade off cut quality in favor of piece compactness isn't just an empirical observation, it is a well understood consequence of the theoretical analysis of spectral partitioning methods.)

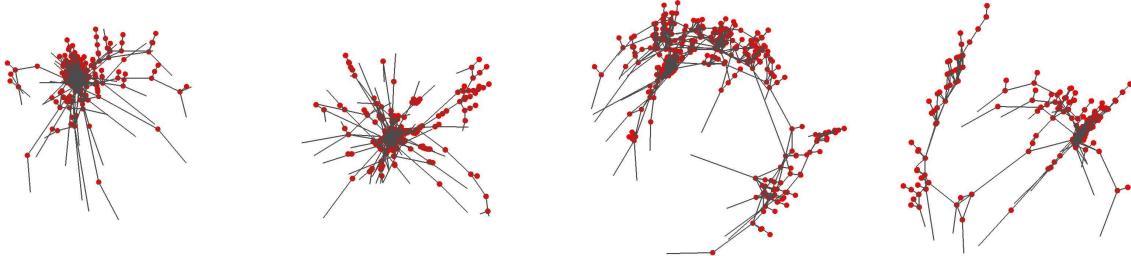


Figure 10.16: Two examples of “communities” found by the LocalSpectral algorithm (on the left) and two from the Metis+MQI algorithm (on the right). Note that the Local Spectral “communities” are more compact—they are tighter and have smaller diameter since the algorithm has difficulty pushing probability mass down long extended paths—while the Metis+MQI “communities” are more sprawling—they have larger diameter and more diverse internal structure, but better conductance scores. In both cases, we have shown communities with ca. 500 nodes (many of which overlap at resolution of this figure), *i.e.*, just above the “whisker” size scale.

Finally, in the bottom row of Figure 10.15 we briefly introduce the topic of internal vs. external cuts, which is something that none of our algorithms are explicitly trying to optimize. These are again scatter plots showing the same set of Local Spectral and Metis+MQI pieces as before, but now the y-axis is external conductance divided by internal conductance. External conductance is the quantity that we usually plot, namely the conductance of the cut which separates the piece from the graph. Internal conductance is the score of a low conductance cut *inside* the piece (that is, in the induced subgraph on the piece’s nodes). Intuitively, good communities should have small ratios, ideally below 1.0, which would mean that they are well separated from the rest of the network, but that they are internally well-connected. However, the three bottom-row plots show that for these three sample graphs, there are mostly no ratios well below 1.0 except at small sizes. (Of course, any given graph could happen to contain a very distinct piece of any size, and the roughly thousand-node piece in the EMAIL-ENRON network is a good example.)

This demonstrates another aspect of our findings: small communities of size below ≈ 100 nodes are internally compact and well separated from the remainder of the network, whereas larger pieces are so hard to separate that separating them from the network is more expensive than separating them internally.

10.6 Models for network community structure

In this section, we use results from previous sections to devise a model that explains the shape of NCP plots. In Section 10.6.1, we examine the NCP plot for a wide range of existing commonly-used network generation models, and we see that none of them reproduces the observed properties, at even a qualitative level. Then, in Section 10.6.2, we analytically demonstrate that certain aspects of the NCP plot, *e.g.*, the existence of deep cuts at small size scales, can be explained by very sparse random graph models. Then, in Section 10.6.3, we present a simple toy model to develop intuition about the effect we must reproduce with a realistic generative model. Finally, in Section 10.6.4, we will combine these and other ideas to describe a Forest Fire graph generation model that reproduces quite well our main observations.

10.6.1 NCP plots for commonly-used network generation models

We have studied a wide range of commonly-used network generative models in an effort to reproduce the upward-sloping NCP plots and to understand the structural properties of the real-world networks that are responsible for this phenomenon. In each case, we have experimented with a range of parameters, and in no case have we been able to reproduce our empirical observations, at even a qualitative level. In Figure 10.17, we summarize these results.

There has been a large body of work subsequent to that of Albert and Barabási [Barabási and Albert, 1999] on models in which edges are added via a preferential-attachment or rich-gets-richer mechanism [Newman, 2003, Bollobas and Riordan, 2003]. Much of this work aims at reproducing properties of real-world graphs such as heavy-tailed degree distributions [Albert et al., 1999, Broder et al., 2000, Faloutsos et al., 1999]. In these preferential attachment models, one typically connects each new node to the existing network by adding exactly m edges to existing nodes with a probability that depends on the current degree of that existing node. Figure 10.17(a) shows the NCP plot for a 10,000 node network generated according to the original preferential attachment model [Barabási and Albert, 1999], where at each time step a node joins the graph and connects to $m = 2$ existing nodes. Note that the NCP plot is very shallow and flat (more even than the corresponding rewired graph), and thus the network that is generated is expander-like at all size scales.

A different type of generative model is one in which edges are added via a copying mechanism [Kumar et al., 2000]. In this copying model, a new node joins the network by attaching exactly m edges to existing nodes as follows: the new node first selects uniformly at random a “seed” or “ambassador” node u ; then, for each of its m edges, with probability β the new node links to an existing node chosen randomly, and with probability $1 - \beta$ it links to a random neighbor of node u . In Figure 10.17(b), we show the results for a network with 50,000 nodes, generated with $m = 2$ and $\beta = 0.05$. Although intuitively the copying model aims to produce communities by linking a new node to neighbors of a existing node, this does not seem to be the right mechanism to reproduce the NCP plot since potential ambassador nodes are all treated similarly and since new nodes always create the same number of edges.

Next, in Figure 10.17(c), we consider an example of a network that was designed to have a recursively hierarchical community structure [Ravasz et al., 2002, Ravasz and Barabási, 2003]. In this model, we start with a 5-node square-like structure with a central node, and then recursively expand the square and link it to the middle node of the network. This network has power-law degree distribution, and clustering coefficient that decays as in a characteristic manner [Ravasz and Barabási, 2003]. In this case, however, the NCP plot is sloping downwards. The local dips in the plot correspond to multiples of the size of the basic module of the graph. Although the model generates links such that nodes that are farther apart in the hierarchy link less frequently, the NCP plot clearly indicates that in aggregate larger communities are easily separated than smaller communities.

A different way to generate hierarchical networks with power-law degree distributions is the Community Guided Attachment model [Leskovec et al., 2005b]. Here we decompose the nodes of a graph into a nested groups of nodes, such that the difficulty of forming links between nodes in different groups increases exponentially with the distance in the community hierarchy. Graphs generated by this principle have both power-law degree distributions and they also obey the Densification Power Law [Leskovec et al., 2005b, 2007b]. As Figure 10.17(d) shows, though, the NCP plot is sloping downward. Qualitatively this plot from CGA is very similar to the plot of the recursive hierarchical construction in Figure 10.17(c), which is not surprising given the similarities of the models.

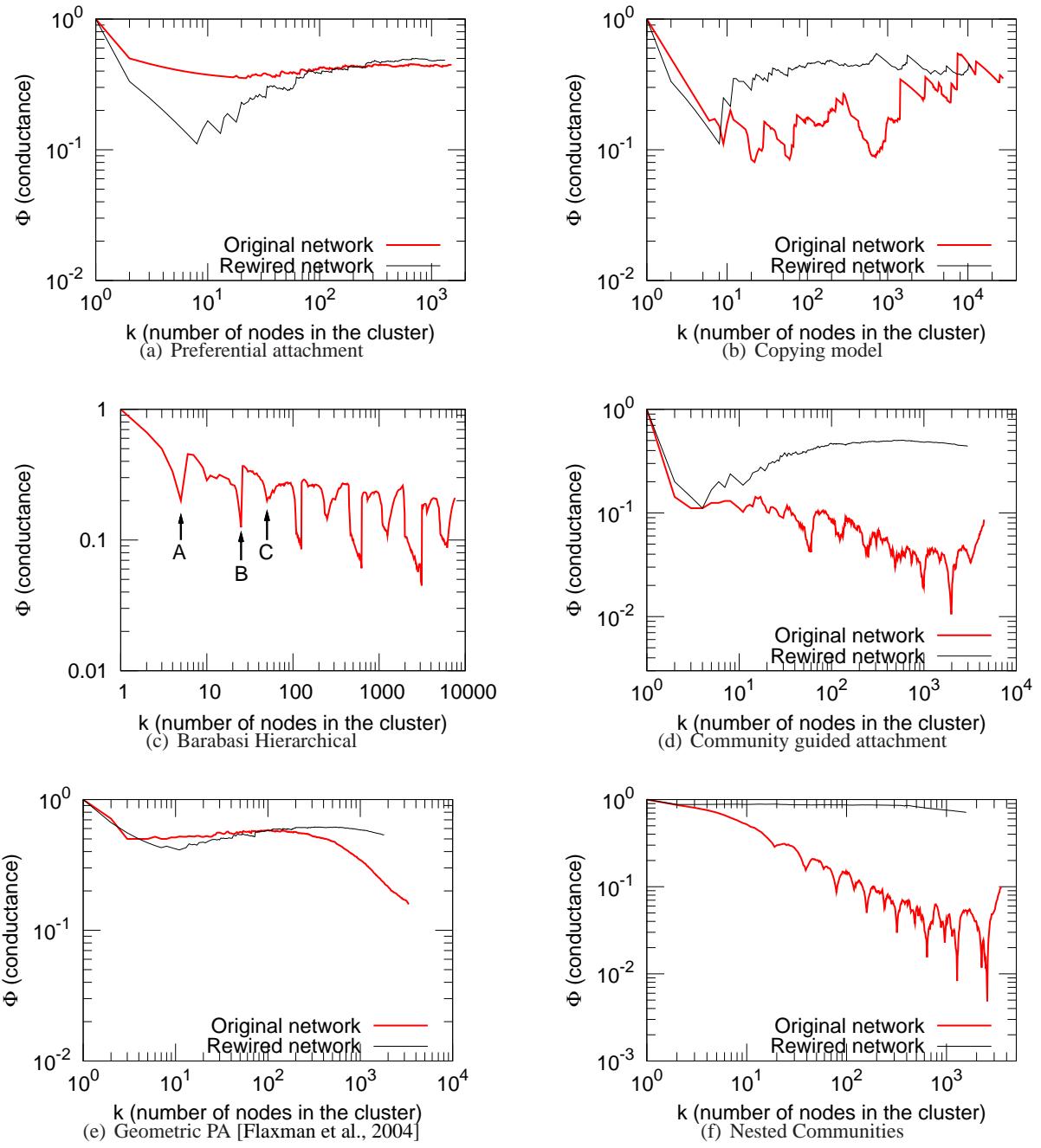


Figure 10.17: Network community profile for networks generated from commonly-used procedures to generate graphs with heavy-tailed degree distributions: (10.17(a)) Preferential attachment; (10.17(b)) Copying model; (10.17(c)) Hierarchical model; (10.17(d)) Community guided attachment; (10.17(e)) Geometric preferential attachment; and (10.17(f)) Nested community model. See the text for details. Red curves plot the results of the Local Spectral Algorithm on the specified network, and black curves plot the results of the Local Spectral Algorithm applied to a randomly rewired version of the same network.

Figure 10.17(e) shows the NCP plot for a geometric preferential attachment model [Flaxman et al., 2004, 2007]. This model aims to achieve a heavy-tailed degree distribution as well edge locality, and it does so by making the connection probabilities depend both on the two-dimensional geometry and on the preferential attachment scheme. As we see, the effect of the underlying geometry eventually dominates the NCP plot since the best bi-partitions are fairly well-balanced [Flaxman et al., 2004]. Intuitively, geometric preferential attachment graphs look locally expander-like, but at larger size scales the union of such small expander graphs behaves like a geometric mesh. We also experimented with the small-world model by Watts and Strogatz [Watts and Strogatz, 1998], in which the NCP plot in some sense behaves exactly the opposite (plot not shown): first the NCP plot decreases, and then it flattens out. Intuitively, a small-world network looks locally like a mesh, but when one reaches larger size scales, the randomly rewired edges start to appear and the graph looks like an expander.

Finally, we explored in more detail networks with explicitly planted community structure. For example, we started with 10 isolated communities generated using the $G_{n,p}$ model, and then we generated a random binary tree. For each internal node at height h we link the nodes in both sides of the tree with probability p^h , for a probability parameter p . This and other related networks gives a graph of nested communities resembling the hierarchical clustering algorithm of Newman and Girvan [Newman and Girvan, 2004]. We see, however, from Figure 10.17(f) that the NCP plot slopes steadily downward, and furthermore we observe that dips correspond to the cuts that separate the communities.

These experiments demonstrate that hierarchically nested networks and networks with underlying geometric or expander like structure exhibit very different NCP plots than observed in real networks. So the question still remains: what causes NCP plot to decrease and then start to increase?

10.6.2 Very sparse random graphs have very unbalanced deep cuts

In this section, we will analyze a very simple random graph model which reproduces relatively deep cuts at small size scales and which has a NCP plot that then flattens out. Understanding why this happens will be instructive as a baseline for understanding the community properties we have observed in our real-world networks.

Here we work with the random graph model with given expected degrees, as described by Chung and Lu [Chung and Lu, 2006a, 2002b, Chung et al., 2003a, Chung and Lu, 2002a, 2003, Chung et al., 2003b, 2004, Chung and Lu, 2006b]. Let n , the number of nodes in the graph, and a vector $\mathbf{w} = (w_1, \dots, w_n)$, which will be the expected degree sequence vector (where we will assume that $\max_i w_i^2 < \sum_k w_k$), be given. Then, in this random graph model, an edge e_{ij} between nodes i and j is added, independently, with probability $p_{ij} = w_i w_j / \sum_k w_k$. Thus, $P(e_{ij} = 1) = p_{ij}$ and $P(e_{ij} = 0) = 1 - p_{ij}$. We use $G(\mathbf{w})$ to denote a random graph generated in this manner. (Note that this model is different than the so-called “configuration model” in which the degree distribution is exactly specified and which was studied by Molloy and Reed [Molloy and Reed, 1995, 1998] and also Aiello, Chung, and Lu [Aiello et al., 2000, 2001]. This model is also different than generative models such as preferential attachment models [Barabási and Albert, 1999, Newman, 2003, Bollobas and Riordan, 2003] or models based on optimization [Doyle and Carlson, 2000, 2002, Fabrikant et al., 2002], although common to all of these generative models is that they attempt to reproduce empirically-observed power-law behavior [Albert et al., 1999, Faloutsos et al., 1999, Broder et al., 2000, Newman, 2005, Clauset et al., 2007].)

In this random graph model, the expected average degree is $w_{av} = \frac{1}{n} \sum_{i=1}^n w_i$ and the expected second-order average degree is $\tilde{w} = \sum_{i=1}^n w_i^2 / \sum_k w_k$. Let $w_G = \sum_i w_i$ denote the expected total degree. Given

a subset S of nodes, we define the volume of S to be $w_S = \sum_{v \in S} w_v$ and we say that S is c -giant if its volume is at least cw_G , for some constant $c > 0$. We will denote the actual degrees of the graph G by $\{d_1, d_2, \dots, d_n\}$, and will define $d(S)$ to be the sum of the actual degrees of the vertices in S . Clearly, by linearity of expectation, for any subset S , $E(d(S)) = w_S$.

The special case of the $G(\mathbf{w})$ model in which \mathbf{w} has a power law distribution is of interest to us here. (The other interesting special case, in which all the expected degrees w_i are equal to np , for some $p \in [0, 1]$, corresponds to the classical Erdős-Renyi G_{np} random graph model [Bollobás, 1985].) Given the number of nodes n , the power-law exponent β , and the parameters w and w_{\max} , Chung and Lu [Chung and Lu, 2006a] give the degree sequence for a power-law graph:

$$w_i = ci^{-1/(\beta-1)} \text{ for } i \text{ s.t. } i_0 \leq i < n + i_0, \quad (10.10)$$

where, for the sake of consistency with their notation, we index the nodes from i_0 to $n + i_0 - 1$, and where $c = c(\beta, w, n)$ and $i_0 = i_0(\beta, w, n, w_{\max})$ are as follows:

$$c = \alpha w n^{1/(\beta-1)} \text{ and } i_0 = n \left(\alpha \frac{w}{w_{\max}} \right)^{\beta-1}, \quad (10.11)$$

where we have defined $\alpha = \frac{\beta-2}{\beta-1}$. It is easy to verify that: $w_{\max} = \max_i w_i$ is the maximum expected degree; the average expected degree is given by $w_{av} = \frac{1}{n} \sum_{i=1}^n w_i = w(1+o(1))$; the minimum expected degree is given by $w_{\min} = \min_i w_i = w\alpha(1-o(1))$; and the number of vertices that have expected degree in the range $(k-1, k]$ is proportional to $k^{-\beta}$.

The following theorem characterizes the shape of the NCP plot for this $G(\mathbf{w})$ model when the degree distribution follows Equation (10.10), with $\beta \in (2, 3)$. The theorem makes two complementary claims. First, there exists at least one (small but moderately deep) cut in the graph of size $\Theta(\log n)$ and conductance $\Theta(\frac{1}{\log n})$. Second, for some constants c' and ϵ , there are no cuts in the graph of size greater than $c' \log n$ having conductance less than ϵ . That is, this model has clusters of logarithmic size with logarithmically deep cuts, and once we get beyond this size scale there do not exist any such deep cuts.

Theorem 10.6.1. *Consider the random power-law graph model $G(\mathbf{w})$, where \mathbf{w} is given by Equation (10.10), where $w > 5.88$, and the power-law exponent β satisfies $2 < \beta < 3$. Then, then with probability $1 - o(1)$:*

1. *There exists a cut of size $\Theta(\log n)$ whose conductance is $\Theta\left(\frac{1}{\log n}\right)$.*
2. *There exists $c', \epsilon > 0$ such that there are no sets of size larger than $c' \log n$ having conductance smaller than ϵ .*

Proof. Combine the results of Lemma 10.6.2 and Lemma 10.6.4. □

The two claims of Theorem 10.6.1 are illustrated in Figure 10.18(a). Note that when $w \geq \frac{4}{e}$ and $\beta \in (2, 3)$ then a typical graph in this model is not fully connected but does have a giant component [Chung and Lu, 2006a]. (The well-studied $G_{n,p}$ random graph model [Bollobás, 1985] has a similar regime when $p \in (1/n, \log n/n)$, as will be discussed in Section 10.7.4.)

In addition, under certain conditions on the average degree and second order average degree, the average distance between nodes is in $O(\log \log n)$ and yet the diameter of the graph is $\Theta(\log n)$. Thus, in

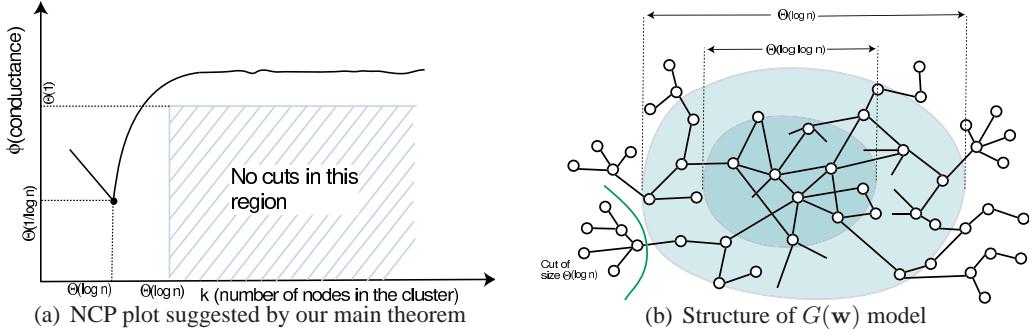


Figure 10.18: The $G(\mathbf{w})$ model in the sparse $\beta \in (2, 3)$ parameter regime. (a) Network community profile plot, as suggested by our main theorem. (b) Caricature of network structure.

this case, the graph has an ‘‘octopus’’ structure, with a subgraph containing $n^{c}/(\log \log n)$ nodes constituting a deep core of the graph. The diameter of this core is $O(\log \log n)$ and almost all vertices are at a distance of $O(\log \log n)$ from this core. However, the pairwise average distance of nodes in the entire graph is $O(\log n / \log \tilde{w})$. A schematic picture of the $G(\mathbf{w})$ model when $\beta \in (2, 3)$ is presented in Figure 10.18(b).

Our first lemma claims that for the $G(\mathbf{w})$ model, if the degree distribution \mathbf{w} follows the above power-law, then there exists a moderately large cut with small conductance. In order to prove the existence of a cut of size $\Theta(\log n)$ and conductance $\Theta(\frac{1}{\log n})$, it is sufficient to concentrate on the existence of whiskers that are large enough. In particular, to prove the following lemma, we compute the probability that there exists a cut of both volume and size $\Theta(\log n)$ and cut-size 1. (Note that although we formally state the lemma in terms of the power-law random graph model, the proof will show that the main claim holds for a more general representation of the heavy-tailed degree distribution.)

Lemma 10.6.2. *For the $G(\mathbf{w})$ model, where w follows a power-law degree distribution with $2 < \beta < 3$ then, with probability $1 - o(1)$ there exists a set of size $\Theta(\log n)$ with conductance $\Theta(\frac{1}{\log n})$.*

Proof. Let S be a subset with the following description. $S = \{v_0, v_1, \dots, v_k\}$, where $k = c_1 \log n$. Let w_i denote the degree of v_i . We have that $w_0 \in [c_2 \log n, 2c_2 \log n]$ and $w_i \leq w$ for all $i > 0$. Thus the expected volume of S is $w_S \in [(2\alpha w c_2 + c_1) \log n, (2\alpha w c_2 + c_1) \log n]$, and the size of S is $c_1 \log n + 1$. Note that the expected volume of the graph can be computed as $w_G = w n$, and hence $\rho = \frac{1}{w_G} = \frac{1}{w n}$.

Now, let n_1 denote the number of vertices of expected degree at most $2\alpha w$. By simple calculation, $n_1 \geq n/2$. The number of possible choices for the vertex v_0 can be computed as follows. Let B be the set of vertices having degree greater than $2\alpha w c_2 \log n$ and A be the set of vertices with degree at most $2\alpha w c_2 \log n$. Then the number of nodes with degree in $[c_2 \log n, 2c_2 \log n]$ is given by the size of $V \setminus (A \cup B)$ which is

$$\alpha w \left(\frac{n}{c_2 \log n} \right)^{\beta-1} - \alpha w \left(\frac{n}{2c_2 \log n} \right)^{\beta-1} \geq \alpha w \left(\frac{n}{c_2 \log n} \right)^{\beta-1},$$

since $\beta > 2$. Thus the number of possible such subsets S is given by the number of choices for v_0 times the number of possible choices for the nodes v_1, \dots, v_k . Thus, the number N of possible such subsets S is at least

$$N = \binom{n_1}{c_1 \log n} \times \alpha \left(\frac{n}{2c_2 \log n} \right)^{\beta-1}.$$

We say that S is good if, after instantiating all the edges, S has a star of size $c_1 \log n$ centered at v_0 , and v_0 is connected to \bar{S} by exactly one edge, and none of the other vertices in S have any edge to \bar{S} . The probability that a particular set S is good is the product of the following terms: the probability p_1 that there is star of size $c_1 \log n$ with v_0 at the center, the probability p_2 that none of the nodes v_1, \dots, v_k link to any nodes in \bar{S} , and the probability p_3 that v_0 connects to \bar{S} using exactly one edge. We now calculate the three probabilities as follows. First,

$$p_1 = \prod_{i \in [1 \dots k]} w_0 w_i \rho \geq (w_0 \alpha w \rho)^{c_1 \log n},$$

since each $w_i \geq w_{\min} \geq \alpha w$. Next,

$$p_2 = \prod_{i=1, \dots, k} \prod_{j \notin S} (1 - w_j \rho) \geq \prod_{i=1, \dots, k} \prod_{j \notin S} e^{-w_j \rho / 2} = e^{-(c_1 \rho 2 \alpha w w_{\bar{S}} \log n) / 2},$$

obtained by using $1 - x \geq e^{-x/2}$ for $0 < x < 1$, and $w_i \leq 2 \alpha w$ for $i \in S, i > 1$. Finally, we get p_3 as follows. First note

$$\begin{aligned} p_3 &= \sum_{j \in \bar{S}} w_0 w_j \rho \prod_{k \neq j, k \in \bar{S}} (1 - w_k w_0 \rho) \\ &\geq \sum_{j \in \bar{S}} w_0 w_j \rho e^{-(w_{\bar{S}} - w_j) w_0 \rho / 2} \\ &= w_0 \rho e^{-w_{\bar{S}} w_0 \rho / 2} \left(\sum_{j \in \bar{S}} w_j e^{w_j w_0 \rho / 2} \right). \end{aligned}$$

Then, since $w_j w_0 \rho \ll 1$ and since $e^x \geq 1 + x$, we have that

$$\begin{aligned} p_3 &\geq w_0 \rho e^{-w_{\bar{S}} w_0 \rho / 2} \left(\sum_{j \in \bar{S}} w_j \left(1 + \frac{w_j w_0 \rho}{2} \right) \right) \\ &\geq w_0 \rho e^{-w_{\bar{S}} w_0 \rho / 2} (w_{\bar{S}} + w_0 \rho \tilde{w}_{\bar{S}} / 2), \end{aligned}$$

where $\tilde{w}_{\bar{S}} = \sum_{j \in \bar{S}} w_j^2$. So the final probability of goodness of S is

$$\begin{aligned} p &= p_1 \times p_2 \times p_3 \geq (w_0 \alpha w \rho)^{c_1 \log n} \times e^{-(c_1 \rho 2 \alpha w w_{\bar{S}} \log n) / 2} \times w_0 \rho e^{-w_{\bar{S}} w_0 \rho / 2} (w_{\bar{S}} + w_0 \rho \tilde{w}_{\bar{S}} / 2) \\ &= (w_0 \alpha w \rho)^{c_1 \log n} \times e^{-(c_1 \gamma 2 \alpha w \log n)} \times w_0 \rho e^{-\gamma w_0} (w_{\bar{S}} + w_0 \rho \tilde{w}_{\bar{S}} / 2), \end{aligned}$$

using $\gamma = \rho w_{\bar{S}} / 2$. So the expected number of such good subsets S is

$$\begin{aligned} Np &\geq \binom{n_1}{c_1 \log n} \times \alpha w \left(\frac{n}{2c_2 \log n} \right)^{\beta-1} \times (w_0 \alpha w \rho)^{c_1 \log n} \times e^{-(c_1 \gamma 2 \alpha w \log n)} \times w_0 \rho e^{-\gamma w_0} (w_{\bar{S}} + w_0 \rho \tilde{w}_{\bar{S}} / 2) \\ &\geq \left(\frac{n_1}{c_1 \log n} \right)^{c_1 \log n} \times \frac{\alpha w n^{\beta-1}}{(2c_2 \log n)^{\beta-1}} \times (w_0 \alpha w \rho)^{c_1 \log n} \times e^{-(c_1 \gamma 2 \alpha w \log n)} \times w_0 \rho e^{-\gamma w_0} \times nw / 2, \end{aligned}$$

using Stirling's formula and the fact that $w_{\bar{S}} \geq nw/2$. Using the value of n_1 and since $nw\rho = 1$,

$$\begin{aligned} Np &\geq \left(\frac{n}{2c_1 \log n} \right)^{c_1 \log n} \times \frac{\alpha w n^{\beta-1}}{(2c_2 \log n)^{\beta-1}} \times (w_0 \alpha w \rho)^{c_1 \log n} \times e^{-(c_1 \gamma 2 \alpha w \log n)} \times e^{-\gamma w_0} \times w_0 / 2 \\ &\geq \left(\frac{w_0 \alpha}{2c_1 \log n} \right)^{c_1 \log n} \times \frac{\alpha w n^{\beta-1}}{(2c_2 \log n)^{\beta-1}} \times (w_0 \alpha w \rho)^{c_1 \log n} \times e^{-(c_1 \gamma 2 \alpha w \log n)} \times e^{-\gamma w_0} \times w_0 / 2. \end{aligned}$$

Using $w_0 \geq c_2 \log n$, we have that have that

$$\begin{aligned} Np &\geq \left(\frac{c_2\alpha}{2c_1}\right)^{c_1 \log n} \times \frac{\alpha w n^{\beta-1}}{2(2c_2 \log n)^{\beta-2}} \times e^{-(c_1 \gamma 2\alpha w \log n)} \times e^{-\gamma w_0} \\ &\geq e^{\Theta \log n} \times \frac{\alpha w}{2(2c_2 \log n)^{\beta-2}}, \end{aligned}$$

where $\Theta = c_1 \log(\frac{c_2\alpha}{2c_1}) + (\beta - 1) - \gamma \alpha w c_1 - 2\gamma c_2$. Note that for $2 < \beta < 3$, we have that $0 < \alpha < \frac{1}{2}$. Also, $\gamma = \frac{1}{2} - o(1)$. Thus, choosing $c_2 = 2ec_1/\alpha$ and $c_1 = \frac{\beta-2}{2\gamma\alpha w + 4\gamma e/\alpha - 1}$, we get $\Theta = 1$. So,

$$Np \geq e^{\log n} \times \frac{\alpha w}{2(2c_2 \log n)^{\beta-2}} = \Omega(\log n)$$

Then, the probability is a particular set S is good is $p \geq \Omega\left(\frac{(\log n)^{\beta-2}}{N}\right)$. Hence the probability of getting a good set is

$$1 - (1 - p)^N \geq 1 - \left(1 - \Omega\left(\frac{(\log n)^{\beta-2}}{N}\right)\right)^N \geq 1 - o(1)$$

□

We next state the well-known Chernoff bound [Chung and Lu, 2006a], which we will use below.

Lemma 10.6.3. *Let $X = \sum_i X_i$ where the X_i are independent random variables with $X_i \geq -M$. Define $\|X\|^2 = \sum_i \mathbf{E}(X_i^2)$. Then,*

$$\Pr(X \geq \mathbf{E}(X) - \lambda) \leq \exp\left(-\frac{\lambda^2}{2(\|X\|^2 + M\lambda/3)}\right). \quad (10.12)$$

Finally, we show that there are no deep cuts with size greater than $\Theta(\log n)$. To state this lemma, define a connected set S to be ϵ -deficit set if it has actual volume $d(S) \leq \frac{1}{2}d(G)$ and if the conductance of the cut (S, \bar{S}) is at most ϵ , i.e., if the number of edges leaving S is at most $\epsilon d(S)$.

Lemma 10.6.4. *For the $G(\mathbf{w})$ model, where w follows a power-law degree distribution with $2 < \beta < 3$, if the average degree w satisfies $w \geq 5.88$, then with probability $1 - o(1)$ there exists constants c', ϵ such that there is no ϵ -deficit set of size more than $c' \log n$.*

Proof. Let $e(S, \bar{S})$ denote the actual number of edges between S and \bar{S} . First we compute the probability that a given set S is ϵ -deficit, that is, S satisfies $e(S, \bar{S}) < \epsilon d(S)$. Let $\delta = \frac{2\epsilon}{1-\epsilon}$. For our case, define the variables $X_{(i,j)} = e_{ij}$ for $(i, j) \in (S, \bar{S})$ and $X_{(i,j)} = -\delta e_{ij}$ for $(i, j) \in (S, S)$. Then the sum $X = \sum X_{(i,j)} = \sum_{(i,j) \in (S, \bar{S})} e_{ij} - \delta \sum_{(i,j) \in (S, S)} e_{ij}$. Note that $e(S, \bar{S}) < \epsilon d(S) \iff X \leq 0$. Using the fact that $\mathbf{E}(e_{ij}) = w_i w_j \rho$, we have $\|X\|^2 = \sum \mathbf{E}(X_{ij}^2) = w_S w_{\bar{S}} \rho + \delta^2 w_S^2 \rho$. Furthermore, exploiting the fact that each $X_i \geq -\delta$, we get that

$$\begin{aligned} \Pr(X \leq 0) &= \Pr(X \leq \mathbf{E}(X) - \mathbf{E}(X)) \\ &\leq \exp\left(-\frac{\mathbf{E}(X)^2}{2(\|X\|^2 + \delta \mathbf{E}(X)/3)}\right) \\ &= \exp\left(-\frac{\rho^2 w_S^2 (w_{\bar{S}} - \delta w_S)^2}{2(w_S \rho (w_{\bar{S}} + \delta^2 w_S) + \delta w_S \rho (w_{\bar{S}} - \delta w_S)/3)}\right). \end{aligned}$$

Cancelling ρw_S from both numerator and denominator,

$$\begin{aligned}\Pr(X \leq 0) &\leq \exp\left(-\frac{\rho w_S(w_{\bar{S}} - \delta w_S)^2}{2(w_{\bar{S}} + \delta^2 w_S + \delta w_{\bar{S}}/3 - \delta^2 w_S/3)}\right) \\ &\leq \exp\left(-\frac{\rho w_S(w_{\bar{S}} - \delta w_S)^2}{2(1 + \delta/3 + 2\delta^2/3)w_{\bar{S}}}\right) \leq \exp\left(-\frac{\rho w_S w_{\bar{S}}(1 - 2\delta w_S/w_{\bar{S}})}{2(1 + \delta/3 + 2\delta^2/3)}\right) \\ &\leq \exp\left(-\frac{\rho w_S w_{\bar{S}}(1 - 2\delta)}{2(1 + \delta/3 + 2\delta^2/3)}\right) \leq \exp(-\rho w_S w_{\bar{S}} A_\delta/2),\end{aligned}$$

where $A_\delta = \frac{1-2\delta}{(1+2\delta/3+2\delta^2/3)}$. So this bounds the probability that a particular set S of size k is ϵ -deficit. We will bound the expected number of such ϵ -deficit subsets of size k . First, let $N_{k,\epsilon,\gamma}$ denote the expected number of ϵ -deficit sets of size k that have expected volume $w_S \leq \gamma w_G$. By linearity of expectation,

$$\begin{aligned}N_{k,\epsilon,\gamma} &\leq \sum_{\substack{S:|S|=k \\ w_S \leq \gamma w_G}} w_{i_1} \dots w_{i_k} w_S^{k-2} \rho^{k-1} \exp(-\rho w_S w_{\bar{S}} A_\delta/2) \\ &\leq \sum_{\substack{S:|S|=k \\ w_S \leq \gamma w_G}} \frac{w_S^{2k-2}}{k^k} \rho^{k-1} \exp((-w_S(1-\gamma)A_\delta)),\end{aligned}$$

where we used the fact that $\gamma = \rho w_{\bar{S}}/2$ and also the AM-GM inequality to say that $\prod_{i \in S} w_i \leq \left(\frac{\sum_{i \in S} w_i}{k}\right)^k$. Now, $F(x) = x^{2k-2} e^{-x A_\delta(1-\gamma)}$ is maximized at $x = \frac{2k-2}{A_\delta(1-\gamma)}$. Thus, the above sum is maximized when $w_S = \frac{2k-2}{A_\delta(1-\gamma)}$. Hence,

$$\begin{aligned}N_{k,\epsilon,\gamma} &\leq \frac{n^k \rho^{k-1}}{k!} \frac{2^{(2k-2)} \cdot (k-1)^{(2k-2)}}{(A_\delta(1-\gamma))^{(2k-2)}} \exp(-2k+2) \\ &\leq \frac{(n\rho)^k}{\rho \sqrt{k}(k/e)^k} \frac{1}{k^k} \frac{2^{(2k-2)} \cdot (k-1)^{(2k-2)}}{(A_\delta(1-\gamma))^{(2k-2)}} \exp(-2k+2).\end{aligned}$$

Using $(1 - \frac{1}{k})^{2k} \leq e^{-2}$, it follows that

$$N_{k,\epsilon,\gamma} \leq \frac{1}{4e\sqrt{k}(k-1)^2} \left(\frac{4}{ewA_\delta^2(1-\gamma)^2}\right)^k.$$

We would like $\sum_{k=c \log n}^{cn} N_{k,\epsilon,\gamma}$ to be $o(1)$, for which we need

$$\frac{4}{ewA_\delta^2(1-\gamma)^2} < 1,$$

which gives a bound on average degree:

$$w \geq \frac{4}{A_\delta^2(1-\gamma)^2 e}.$$

For sets of volume $w_S \geq \gamma w_G$, we have the following. From the double-sided Chernoff bound, for any fixed set S ,

$$|w_S - d(S)| \leq \lambda \text{ with probability } 1 - 2 \exp\left(-\frac{\lambda^2}{2(w_S + \lambda/3)}\right).$$

So if $\lambda = \sqrt{w_S} \log n$, we have the above statement with probability $1 - 2 \exp(-3 \log^2 n / 8)$. Similarly,

$$|e(S, \bar{S}) - \mathbf{E}(e(S, \bar{S}))| \leq \lambda \text{ with probability } 1 - 2 \exp\left(-\frac{\lambda^2}{2(\rho w_S w_{\bar{S}} + \lambda/3)}\right).$$

By having $\lambda = \sqrt{\rho w_S w_{\bar{S}}} \log n$ the above probability becomes $1 - 2 \exp(-3 \log^2 n / 8)$. Now, if both these events occur, then the conductance of the set S is at least $1/3$. So the only way we can get an ϵ -deficit set is at by having one of these conditions to be invalid. The total number of sets of expected volume γw_G is bounded by $\binom{w_G}{\gamma w_G}$. So, the expected number of ϵ -deficit sets of volume at least γw_G is bounded by

$$\sum_{\gamma \leq \theta \leq 1/2} \binom{w_G}{\theta w_G} 4 \exp(-3 \log^2 n / 8) \leq \int_{\gamma \leq \theta \leq 1/2} \frac{1}{\sqrt{\theta w_G}} \left(\frac{1}{\theta}\right)^{\theta w_G} 4 \exp(-3 \log^2 n / 8) \leq o(1).$$

Thus, putting the two bounds together, the expected number of ϵ -deficit sets of size greater than $c \log n$ is at most $o(1)$. Thus with probability $1 - o(1)$ there does not exist an ϵ -deficit set of size greater than $c \log n$. \square

10.6.3 An intuitive toy model for generating an upward-sloping NCP plot

We have seen that commonly-studied models, including preferential attachment models, copying models, simple hierarchical models, and models in which there is an underlying mesh-like or manifold-like geometry are not the right way to think out the community structure of large social and information networks. We have also seen that the extreme sparsity of the networks, coupled with randomness, can be responsible for the deep cuts at small scales.

To build intuition as to what the gradually increasing NCP plot might mean, consider Figure 10.19. This is a toy example of a network construction in which the NCP plot has a deep dip at a small size scale and then steadily increases. The network shown in Figure 10.19(a) is an infinite tree that has two parts. The top part, a subtree (with one node in this example, but more generally consisting of n_T nodes) is indicative of the whiskers, or the “small scale” structure of the graph. The remaining tree has the property that the number of children increases monotonically with the level of the node. This property is indicative of the fact that as the size of a cluster grows, the number of neighbors that it has also increases. The key insight in this construction is that the best conductance cuts first cut at the top of the growing tree and then gradually work their way “down” the tree, starting with the small subtrees and moving gradually down the levels, as depicted in Figure 10.19(a).

Thus, intuitively, one can think of small well-separated communities—those below the n_T size scale that consist of subsets of the small trees—starting to grow, and as they pass the n_T size scale and become bigger and bigger, they blend in more and more with the central part of the network, which (since it exhibits certain expander-like properties) does not have particularly well-defined communities. Note (more generally) that if there are n_T nodes in the small tree at the top of the graph, then the dip in the NCP plot in Figure 10.19(b) is of depth $2/(n_T + 1)$. In particular, if $n_T = \Theta(\log n)$ then the depth of this cut is $\Theta(1/\log n)$.

Intuitively, the NCP plot increases since the “cost” per edge for every additional edge inside a cluster increases with the size of the cluster. For example, in cut A in Figure 10.19(a), the “price” for having 3 internal edges is to cut 6 edges, *i.e.*, 2 edges cut per edge inside. To expand the cluster by just a single edge, one has to move one level down in the tree (toward the cut B) where now the price for a single edge is 4 edges, and so on.

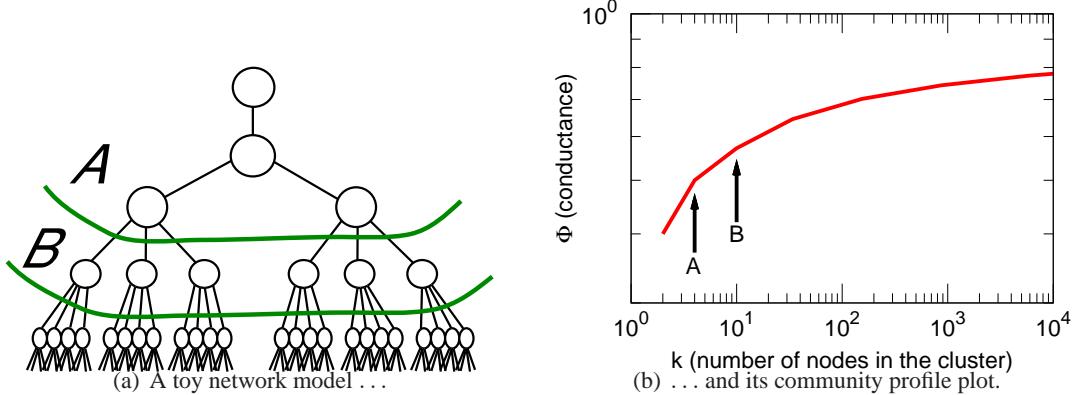


Figure 10.19: Schematic picture of the properties of a network responsible for the upward-sloping community profile plot. (a) This toy model is designed so that the optimal conductance cuts are achieved by cutting nodes from the top of the tree. (b) The minimum of the NCP plot is achieved by cutting the single top node, and then larger and larger cuts have gradually worse and worse conductance values.

10.6.4 A more realistic model of network community structure

The question arises now as to whether we can find a simple generative model that can explain both the existence of small well-separated whisker-like clusters and also an expander-like core whose best clusters get gradually worse as the purported communities increase in size. Intuitively, a satisfactory network generation model must successfully take into account the following two mechanisms:

- (a) The model should produce a relatively large number of relatively small—but still large when compared to random graphs—well connected and distinct whisker-like communities. (This should reproduce the downward part of the community profile plot and the minimum at small size scales.)
- (b) The model should produce a large expander-like core, which may be thought of as consisting of intermingled communities, perhaps growing out from the whisker-like communities, the boundaries of which get less and less well-defined as the communities get larger and larger and as they gradually blend in with rest of the network. (This should reproduce the gradual upward sloping part of the community profile plot.)

The so-called *Forest Fire Model* [Leskovec et al., 2005b, 2007b] captures exactly these two competing phenomena. The Forest Fire Model is a model of graph generation (that generates directed graphs—an effect we will ignore) in which new edges are added via a recursive “burning” mechanism in an epidemic-like fashion. Since the details of the recursive burning process are critical for the model’s success, we explain it in some detail.

To describe the Forest Fire Model of [Leskovec et al., 2005b, 2007b], let us fix two parameters, a *forward burning probability* p_f and a *backward burning probability* p_r . We start the entire process with a single node, and at each time step $t > 1$, we consider a new node v that joins the graph G_t constructed thus far. The node v forms out-links to nodes in G_t as follows:

- (i) Node v first choose a node w , which we will refer to as a “seed” node or an “ambassador” node, uniformly at random and forms a link to w .

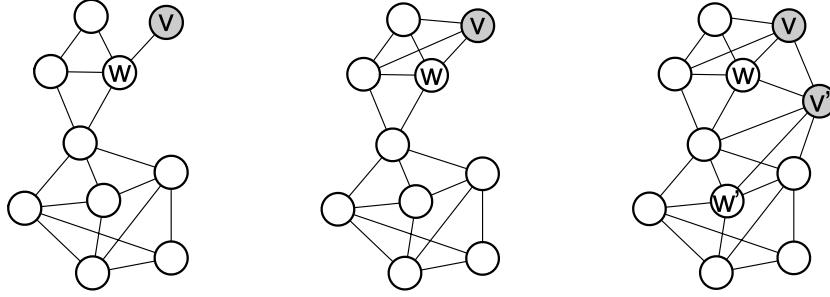


Figure 10.20: The Forest Fire burning process. Left: a new node v joins the network and selects a seed node w . Middle: v then attaches itself by recursively linking to w 's neighbors, w 's neighbor-neighbors, and so on, according to the “forest fire” burning mechanism described in the text. Right: a new node v' joins the network, selects seed w' , and recursively adds links using the same “forest fire” burning mechanism. Notice that if v' causes a large “fire,” it links to a large number of existing nodes. In this way, as potential communities around node w grow, the NCP plot is initially decreasing, but then larger communities around w gradually blend-in with the rest of the network, which leads the NCP plot to increase.

- (ii) Node v selects x out-links and y in-links of w that have not yet been visited. (x and y are two geometrically distributed random numbers with means $p_f/(1-p_f)$ and $p_r/(1-p_r)$, respectively. If not enough in-links or out-links are available, then v selects as many as possible.) Let w_1, w_2, \dots, w_{x+y} denote the nodes at the other ends of these selected links.
- (iii) Node v forms out-links to w_1, w_2, \dots, w_{x+y} , and then applied step (ii) recursively to each of the w_1, w_2, \dots, w_{x+y} , except that nodes cannot be visited a second time during the process.

Thus, burning of links in the Forest Fire Model begins at node w , spreads to w_1, w_2, \dots, w_{x+y} , and proceeds recursively until the process dies out. One can view such a process intuitively as corresponding to a model in which a person comes to the party and first meets an ambassador who then introduces him or her around. If the person creates a small number of friendships these will likely be from the ambassadors “community,” but if the person happens to create many friendships then these will likely go outside the ambassador’s circle of friends. This way, the ambassador’s community might gradually get intermingled with the rest of the network.

Two properties of this model are particularly significant. First, although many nodes might form one or a small number of links, certain nodes can produce large conflagrations, burning many edges and thus forming a large number of out-links before the process ends. Such nodes will help generate a skewed out-degree distribution, and they will also serve as “bridges” that connect formerly disparate parts of the network. This will help make the NCP plot gradually increase. Second, there is a locality structure in that as each new node v arrives over time, it is assigned a “center of gravity” in some part of the network, *i.e.*, at the ambassador node w , and the manner in which new links are added depends sensitively on the local graph structure around node w . Not only does the probability of linking to other nodes decrease rapidly with distance to the current ambassador, but because of the recursive process regions with a higher density of links tend to attract new links.

Figure 10.20 illustrates this. Initially, there is a small community around node w . Then, node v joins and using the Forest Fire mechanism locally attaches to nodes in the neighborhood of seed node w . The

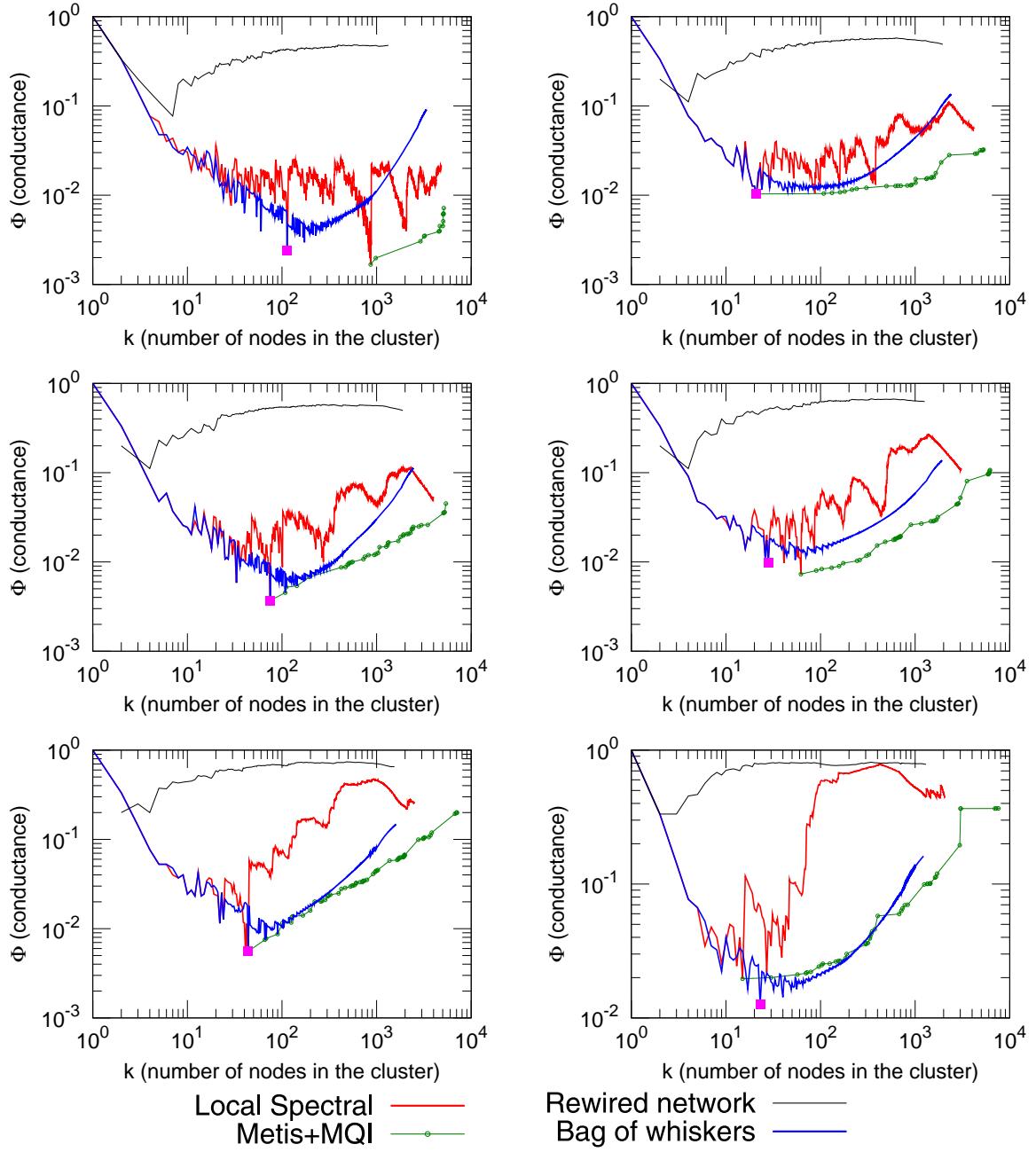


Figure 10.21: Community profile plots for the Forest Fire Model at various parameter settings. The backward burning probability is $p_b = 0.3$, and we increase (left to right, top to bottom) the forward burning probability $p_f = \{0.26, 0.31, 0.33, 0.35, 0.37, 0.40\}$. Note that the largest and smallest values for p_f lead to less realistic community profile plots, as discussed in the text.

growth of the community around w corresponds to downward part of the NCP plot. However, if a node v' then joins and causes a large fire, this has the effect of larger and larger communities around w blending into and merging with the rest of the network.

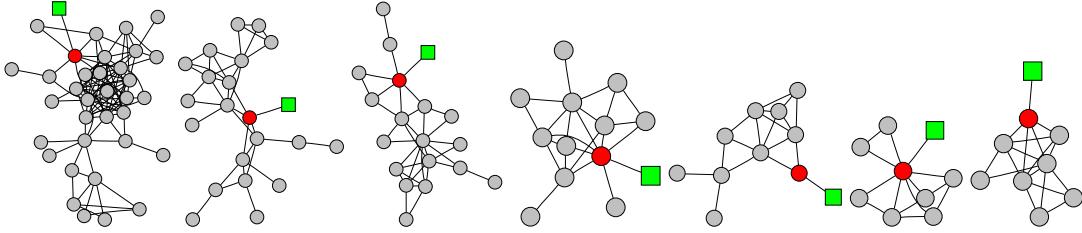


Figure 10.22: Examples of whiskers from a simulation of the Forest Fire Model with parameter settings $p_f = 0.37$ and $p_b = 0.3$. The green square node belongs to the network core, and by cutting the edge connecting it with red circular node we separate the community of circles from the rest of the network (depicted as a green square).

Not surprisingly, however, the Forest Fire Model is sensitive to the choice of the burning probabilities p_f and p_b . We have experimented with a wide range of network sizes and values for these parameters, and in Figure 10.21, we show the community profile plots of several 10,000 node Forest Fire networks generated with $p_b = 0.3$ and several different values of p_f . The first thing to note is that since we are varying p_f the six plots in Figure 10.21, we are viewing networks with very different densities. Next, notice that if, *e.g.*, $p_f = 0.33$ or $p_f = 0.35$ then we observe a very natural behavior: the conductance nicely decreases, reaches the minimum somewhere between 10 and 100 nodes, and then slowly but not too smoothly increases. Not surprisingly, it is in this parameter region where the Forest Fire Model has been shown to exhibit realistic time evolving graph properties such as densification and shrinking diameters [Leskovec et al., 2005b, 2007b].

Next, also notice that if p_f is too low or too high, then we obtain qualitatively different results. For example, if $p_f = 0.26$, then the community profile plot gradually decreases for nearly the entire plot. For this choice of parameters, the Forest Fire does not spread well since the forward burning probability is too small, the network is extremely sparse and is tree-like with just a few extra edges, and so we get large well separated “communities” that get better as they get larger. On the other hand, when burning probability is too high, *e.g.*, $p_f = 0.40$, then the NCP plot has a minimum and then rises extremely rapidly. For this choice of parameters, if a node which initially attached to a whisker successfully burns into the core, then it quickly establishes many successful connections to other nodes in the core. Thus, the network has relatively large whiskers that failed to establish such a connection and a very expander-like core, with no intermediate region, and the increase in the community profile plot is quite abrupt.

We have examined numerous other properties of the graphs generated by the Forest Fire Model and have found them to be broadly consistent with the social and information networks we have examined. One property, however, that is of particular interest is what the whiskers look like. Figure 10.22 shows an example of several whiskers generated by the Forest Fire Model if we choose $p_b = 0.30$ and $p_f = 0.37$. They are larger and more well-structured than the tree-like whiskers from the random graph model of Section 10.6.2. Also notice that they all look plausibly community-like with a core of the nodes densely linked among themselves and the bridge edge then connects the whisker to the rest of the network.

We conclude by noting that there has also been interest in developing hierarchical graph generation models, *i.e.*, models in which a hierarchy is given and the linkage probability between pairs of nodes decreases as a function of their distance in the hierarchy [Ravasz et al., 2002, Ravasz and Barabási, 2003, Chakrabarti et al., 2004, Abello, 2004, Leskovec et al., 2005a, Clauset et al., 2006, Xuan et al., 2006, Leskovec and Faloutsos, 2007]. The motivation for this comes largely from the intuition that nodes in social networks and are joined in to small relatively tight groups that are then further join into larger

groups, and so on. As Figures 10.17(c) and 10.17(d) make clear, however, such models do not immediately lead to community structure similar to what we have observed and which has been reproduced by the Forest Fire Model. On the other hand, although there are significant differences between hierarchical models and the Forest Fire Model, [Leskovec et al., 2005b, 2007b] notes that there are similarities. In particular, in the Forest Fire Model a new node v is assigned an ambassador w as an entry point to the network. This is analogous to a child having a parent in the hierarchy which helps to determine how that node links to the remainder of the network. Similarly, many hierarchical models have a connection probability that decreases exponentially in the hierarchical tree distance. In the Forest Fire Model, the probability that a node v will burn along a particular path to another node u' is exponentially small in the path length, although the analogy is not perfect since there may exist many possible paths.

10.7 Discussion

In this section, we discuss several aspects of our main results in a broader context. In particular, in Section 10.7.1, we compare to several data sets in which there is some notion of “ground truth” community and we also describe several broader non-technical implications of our results. Then, in Section 10.7.3, we describe recent work on community detection and identification. Finally, in Section 10.7.4, we discuss several technical and algorithmic issues and questions raised by our work.

10.7.1 Comparison with “ground truth” and sociological communities

In this subsection, we examine the relationship between network communities of the sort we have been discussing so far and some notion of “ground truth.” When considering a real network, one hopes that the output of a community finding algorithms will be “real” communities that exist in some meaningful sense in the real world. For example, in the Karate club network in Figure 10.5(a), the cut found by the algorithm corresponds in some sense to a true community, in that it splits the nodes almost precisely as they split into two newly formed karate clubs. In this section, we take a different approach: we take networks in which there are explicitly defined communities, and we examine how well these communities are separated from the rest of the network. In particular, we examine a minimum conductance profile of several network datasets, where we can associate with each node one or more community labels which are exogenously specified. Note that we are overloading the term “community” here, as in this context the term might mean one of two things: first, it can refer to groups of nodes with good conductance properties; second, it can refer to groups of nodes that belong to the same self-defined or exogenously-specified group.

We consider the following five datasets:

- **LIVEJOURNAL12** [Backstrom et al., 2006]: LiveJournal is an on-line blogging site where users can create friendship links to other users. In addition, users can create groups which other users can then join. In LiveJournal, there are 385,959 such groups, and a node belongs to 3.5 groups on the average. Thus, in addition to the information in the interaction graph, we have labels specifying those groups with which a user is associated, and thus we may view each such group as determining a “ground truth” community.
- **CA-DBLP** [Backstrom et al., 2006]: We considered a co-authorship network in which nodes are authors and there is an edge if authors co-authored at least one paper. Here, publication venues (*e.g.*, journals and conferences) can play the role of “ground truth” communities. That is, an author is a

member of a particular group or community if he or she published at a particular conference or in a particular journal. In our DBLP network, there are 2,547 such groups, with a node belonging to 2.6 on the average.

- AMAZONALLPROD [Clauset et al., 2004]: This is a network of products that are commonly purchased together at amazon.com. (Intuitively one might expect that, *e.g.*, gardening books are frequently purchased together, so the network structure might reflect a well-connected cluster of gardening books.) Here, each item belongs to one or more hierarchically organized categories (book, movie genres, product types, etc.), and products from the same category define a group which we will view as a “ground truth” community. Items can belong to 49,732 different groups, and each item belongs to 14.3 groups on the average.
- ATM-IMDB: This network is a bipartite actors-to-movies network composed from IMDB data, and an actor A is connected to movie B if A appeared in B . For each movie we also know the language and the country where it was produced. Countries and languages may be taken as “ground truth” communities or groups, where every movie belongs to exactly one group and actors belongs to all groups to which movies that they appeared in belong. In our dataset, we have 393 language groups and 181 country groups.
- EMAIL-INSIDE and EMAIL-INOUT [Leskovec et al., 2007b]: This is an email communication network from a large European research organization conducting research in natural sciences: physics, chemistry, biology and computer science. Each of 986 members of the organization belongs to exactly one of 45 departments, and we use the department memberships to define “ground truth” communities.

Although none of these notions of “ground truth” is perfect, many community finding algorithms use precisely this form of anecdotal evaluation: a network is taken, network communities are found, and then the correspondence of network communities to “ground truth” communities is evaluated. Note, in contrast, we are evaluating how “ground truth” communities behave at different size scales with respect to our methodology, rather than examining how the groups we find relate to “ground truth” communities. Furthermore, note that the notions of “ground truth” are not all the same—we might expect that people publish papers across several different venues in a very different way than actors appear in movies from different countries. More detailed statistics for each of these networks may be found in Tables A.2, A.3 and A.4.

To examine the quality of “ground truth” communities in the these network datasets, we take all groups and measure the conductance of the cut that separates that group from the rest of the network. Thus, we generated NCP plots in the following way. For every “ground truth” community, we measure the conductance of the cut separating it from the rest of the graph, from which we obtain a scatter plot of community size versus conductance. Then, we take the lower-envelope of this plot, *i.e.*, for every integer k we find the conductance value of the community of size k that has the lowest conductance. Figure 10.23 shows the results for these network datasets; the figure also shows the NCP plot obtained from using the Local Spectral Algorithm on both the original network (plotted in red) and on the rewired network (plotted in black).

Several observations can be made:

- The conductance of “ground truth” communities follows that for the network communities up to until size 10-100 nodes, *i.e.*, larger communities get successively more community-like. As “ground truth” communities get larger, their conductance values tend to get worse and worse, in agreement

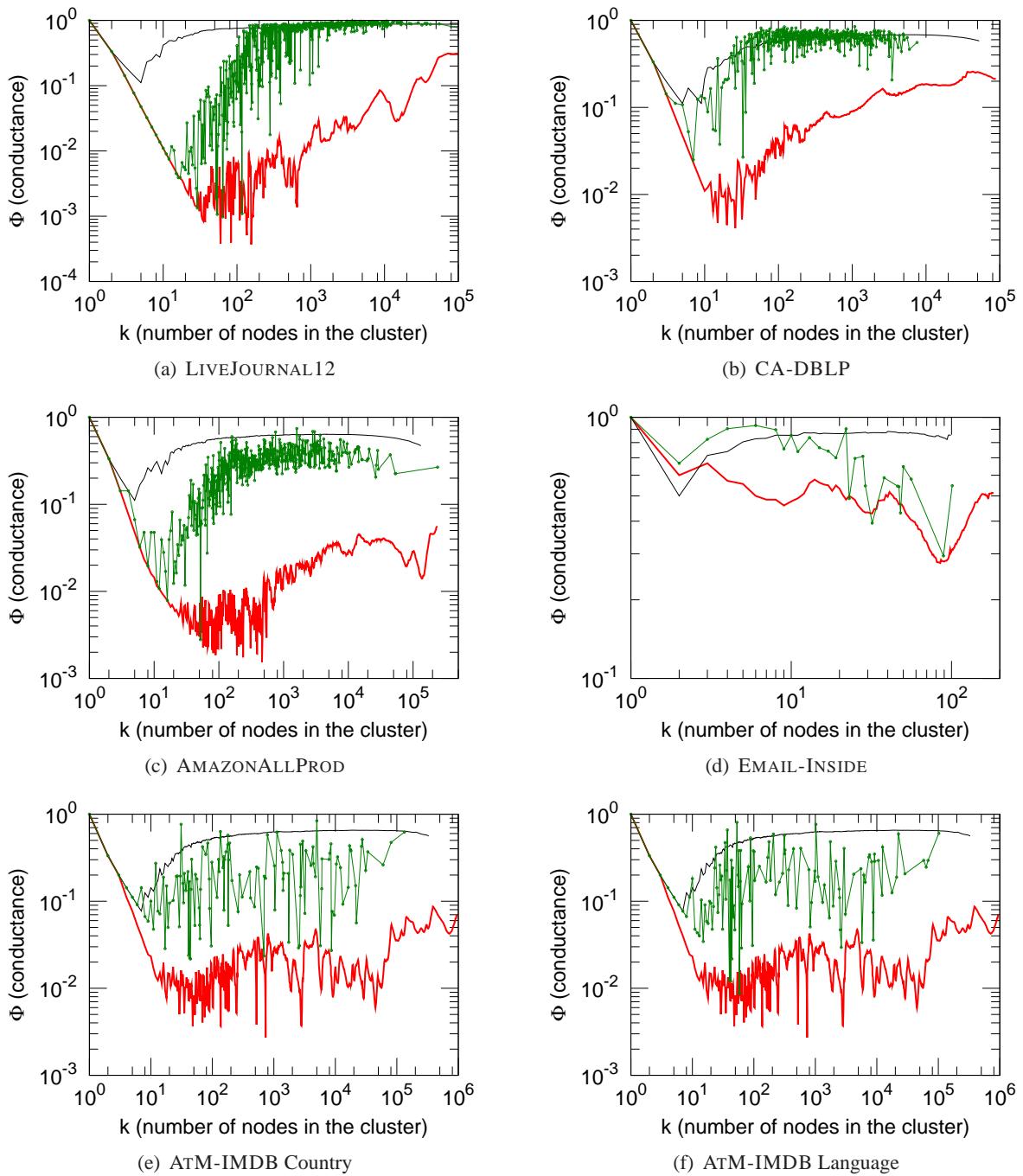


Figure 10.23: Network community profile plots for explicitly “ground truth” communities (green), compared with that for the original network (red) and a rewired version of the network (black): (a) LIVEJOURNAL12; (b) CA-DBLP; (c) AMAZONALLPROD; (d) EMAIL-INSIDE; and (e-f) ATM-IMDB.

with network communities discovered with graph partitioning approximation algorithms. Thus, the qualitative trend we observed in nearly every large sparse real-world network (of the best communities blending in with the rest of the network as they grow in size) is seen to hold for small “ground truth” communities.

- One might expect that the NCP plot for the “ground truth” communities (the green curves) will be somewhere between the NCP plot of the original network (red curves) and that for the rewired network (black curves), and this is seen to be the case in general. The NCP plot for network communities goes much deeper and rises more gradually than for “ground truth” communities. This is also consistent with our general observation that only small communities tend to be dense and well separated, and to separate large groups one has to cut disproportionately many edges.
- For the two social networks we studied (LIVEJOURNAL12 and CA-DBLP), larger “ground truth” communities have conductance scores that get quite “random”, *i.e.*, they are as well separated as they would be in a randomly rewired network (green and black curves overlap). This is likely associated with the relatively weak and overlapping notion of “ground truth” we associated with those two network datasets. On the other hand, for AMAZONALLPROD and ATM-IMDB networks, the general trend still remains but large “ground truth” communities have conductance scores that lie well below the rewired network curve.

Our email network illustrates a somewhat different point. The NCP plot for EMAIL-INSIDE should be compared with that for EMAIL-INOUT, which is displayed in Figure 10.7. The EMAIL-INSIDE email network is rather small, and so it has a decreasing community profile plot, in agreement with the results for small social networks. Since communication is mainly focused between the members of the same department, both network and “ground truth” communities are well expressed. Next, compare the NCP plot of EMAIL-INSIDE with that of EMAIL-INOUT (Figure 10.7). We see that the NCP plot of EMAIL-INSIDE slopes downwards (as we consider only the communication inside the organization), but as soon as we consider the communication inside the organization and to the outside world (EMAIL-INOUT, or alternatively, see EMAIL-ENRON) then we see a completely different and more familiar picture—the NCP plot drops and then slowly increases. This suggests that the organizational structure, (*e.g.*, departments) manifest themselves in the internal communication network, but as soon as we put the organization into the broader context (*i.e.*, how it communicates to the rest of the world) then the internal department structure seems to disappear.

10.7.2 Connections and broader implications

In contrast to numerous studies of community structure, we find that there is a natural size scale to communities. Communities are relatively small, with sizes only up to about 100 nodes. We also find that above size of about 100, the “quality” of communities gets worse and worse and communities more and more “blend into” the network. Eventually, even the existence of communities (at least when viewed as sets with stronger internal than external connectivity) is rather questionable. We show that large social and information networks can be decomposed into a large number of small communities and a large dense and intermingled network “core”—we empirically establish that the “core” contains on average 60% of the nodes and 80% of all edges. But, as demonstrated by Figure 10.13, the “core” itself has a non-trivial structure—in particular, it has a core-whisker structure that is analogous to the original complete network.

The Dunbar number: Our observation on the limit of community size agrees with Dunbar [Dunbar, 1998] who predicted that roughly 150 is the upper limit on the size of a well-functioning human community. Moreover, Allen [Allen, 2004] gives evidence that on-line communities have around 60 members, and on-line discussion forums start to break down at about 80 active contributors. Church congregations, military companies, divisions of corporations, all are close to the sum of 150 [Allen, 2004]. We are thus led to ask: Why, above this size, is community quality inversely proportional to its size? And why are NCP plots of small and large networks so different?

Previous studies mainly focused on small networks (*e.g.*, see [Danon et al., 2005]), which are simply not large enough for the clusters to gradually blend into one another as one looks at larger size scales. Our results do not disagree with literature at small sizes. But it seems that in order to make our observations one needs to look at large networks. It is only when Dunbar’s limit is passed that we find large communities blurring and eventually vanishing. A second reason is that previous work did not measure and examine the *network community profile* of cluster size vs. cluster quality.

Common bond vs. common identity communities: Dunbar’s explanation aligns well with the common bond vs. common identity theory of group attachment [Ren et al., 2007] from social psychology. Common identity theory makes predictions about people’s attachment to the group as a whole, while common bond theory predicts people’s attachment to individual group members. The distinction between the two refers to people’s different reasons for being in a group. Because they like the group as a whole we get identity-based attachment, or because they like individuals in the group we get bond-based attachment. Anecdotally, bond-based groups are based on social interaction with others, personal knowledge of them, and interpersonal attraction to them. On the other hand, identity-based groups are based on common identity of its members, *e.g.*, liking to play a particular on-line game, contributing to Wikipedia, etc. It has been noted that bond communities tend to be smaller and more cohesive [Back, 1951], as they are based on interpersonal ties, while identity communities are focused around common theme or interest. See [Ren et al., 2007] for a very good review of the topic.

Translating this to our context, the bond vs. identity communities mean that small, cohesive and well-separated communities are probably based on common bonds, while bigger groups may be based on common identity, and it is hard to expect such big communities to be well-separated or well-expressed in a network sense. This further means the transition between common bond (*i.e.*, maintaining close personal ties) and common identity (*i.e.*, sharing a common interest or theme) occurs at around one hundred nodes. It seems that at this size the cost of maintaining bond ties becomes too large and the group either dies or transitions into a common identity community. It would be very interesting as a future research topic to explore differences in community network structure as the community grows and transitions from common bond to common identity community.

Edge semantics: Another explanation could be that in small, carefully collected networks, the semantics of edges is very precise while in large networks we know much less about each particular edge, *e.g.*, especially when online people have very different criteria for calling someone a friend. Traditionally scientists through surveys “normalized” the links by making sure each link has the same semantics/strength.

Evidence in previous work: There has also been some evidence that hints towards the findings we make here. For example, Clauset *et al.* [Clauset et al., 2004] analyzed community structure of a graph related to the AMAZONALLPROD, and they found that around 50% of the nodes belonged to the largest “miscellaneous” community. This agrees with the typical size of the network core, and one could conclude that the largest community they found corresponds to the intermingled network core, and most of the rest of the communities are whisker-like.

In addition, recently there have been several works hinting that the network communities subject is more complex than it seems at the first sight. For example, it has been found that even random graphs can have good modularity scores [Guimerà et al., 2004]. Intuitively, random graphs have no community structure, but there can still exist sets of nodes with good community scores, at least as measured by modularity (due to random fluctuations about the mean). Moreover, very recently a study of robustness of community structure showed that the canonical example of presence of community structure in networks [Zachary, 1977] may have no significant community structure [Karrer et al., 2008].

More general thoughts: Our work also raises an important question of what is a natural community size and whether larger communities (in a network sense) even exist. It seems that when community size surpasses some threshold, the community becomes so diverse that it stops existing as a traditionally understood “network community.” Instead, it blends in with the network, and intuitions based on connectivity and cuts seem to fail to identify it. Approaches that consider both the network structure and node attribute data might help to detect communities in these cases.

Also, conductance seems like a very reasonable measure that satisfies intuition about community quality, but we have seen that if one only worries about optimizing conductance, then bags of whiskers and other internally disconnected and sparsely connected sets have the best scores. This raises interesting questions about cluster compactness, regularization, and smoothness: what is a good definition of compactness, what is the best way to regularize these noisy networks, and how should this be connected to the notion of community separability?

A common assumption is that each node belongs to exactly one community. Our approach does not make such an assumption. Instead, for each given size, we independently find best set of nodes, and “communities” of different sizes often overlap. As long there is a boundary between communities (even if boundaries overlap), cut- and edge-density- based techniques (like modularity and conductance) may have the opportunity to find those communities. However, it is the absence of clear community boundaries that makes the NCP plot go upwards.

10.7.3 Relationship with community identification methods

A great deal of work has been devoted to finding communities in large networks, and much of this has been devoted to formalizing the intuition that a community is a set of nodes that has more and/or better intra-linkages between its members than inter-linkages with the remainder of the network. Very relevant to our work is that of Kannan, Vempala, and Vetta [Kannan et al., 2004], who analyze spectral algorithms and describe a community concept in terms of a bicriterion depending on the conductance of the communities and the relative weight of inter-community edges. Flake, Tarjan, and Tsioutsiouliklis [Flake et al., 2003] introduce a similar bicriterion that is based on network flow ideas, and Flake *et al.* [Flake et al., 2000, 2002] defined a community as a set of nodes that has more intra-edges than inter-edges. Similar edge-counting ideas were used by Radicchi *et al.* [Radicchi et al., 2004] to define and apply the notions of a strong community and a weak community.

Within the “complex networks” community, Girvan and Newman [Girvan and Newman, 2002] proposed an algorithm that used “centrality” indices to find community boundaries. Following this, Newman and Girvan [Newman and Girvan, 2004] introduced *modularity* as *a posteriori* measure of the strength of community structure. Modularity measures inter- (and not intra-) connectivity, but it does so with reference to a randomized null model. Modularity has been very influential in the recent community detection literature [Newman, 2004, Danon et al., 2005], and one can use spectral techniques to approxi-

mate it [White and Smyth, 2005, Newman, 2006b]. On the other hand, Guimerà, Sales-Pardo, and Amaral [Guimerà et al., 2004] and Fortunato and Barthélémy [Fortunato and Barthélémy, 2007] showed that random graphs have high-modularity subsets and that there exists a size scale below which communities cannot be identified. In part as a response to this, some recent work has had a more statistical flavor [Hastings, 2006, Reichardt and Bornholdt, 2007, Rosvall and Bergstrom, 2007, Airoldi et al., 2007, Karrer et al., 2008, Newman and Leicht, 2007]. In light of our results, this work seems promising, both due to potential “overfitting” issues arising from the extreme sparsity of the networks, and also due to the empirically-promising regularization properties exhibited by local spectral methods.

We have made extensive use of the Local Spectral Algorithm of Andersen, Chung, and Lang [Andersen et al., 2006]. Similar results were originally proven by Spielman and Teng [Spielman and Teng, 2004], who analyzed local random walks on a graph; see Chung [Chung, 2007a,c,b] for an exposition of the relationship between these methods. Andersen and Lang [Andersen and Lang, 2006] showed that these techniques can find (in a scalable manner) medium-sized communities in very large social graphs in which there exist reasonably well-defined communities. In light of our results, such methods seem promising more generally. Other recent work that has focused on developing local and/or near-linear time heuristics for community detection include [Clauset et al., 2004, Wu and Huberman, 2004a, Clauset, 2005, Bagrow and Bollt, 2005, Raghavan et al., 2007].

In addition to this work we have cited, there exists work which views communities from a very different perspective. For example, Kumar *et al.* [Kumar et al., 1999b] view communities as a dense bipartite subgraph of the Web; Gibson, Kleinberg, and Raghavan [Gibson et al., 1998] view communities as consisting of a core of central authoritative pages linked together by hub pages; Hopcroft *et al.* [Hopcroft et al., 2003, 2004] are interested in the temporal evolution of communities that are robust when the input data to clustering algorithms that identify them are moderately perturbed; and Palla *et al.* [Palla et al., 2005] view communities as a chain of adjacent cliques and focus on the extent to which they are nested and overlap. The implications of our results for this body of work remain to be explored.

10.7.4 Relationship with other theoretical work

In this section, we describe the relationship between technical aspects of graph partitioning and finding good cuts in our work and the recent work with similar flavor in graph partitioning, algorithms, and graph theory.

Recent work has focused on the expansion properties of power law graphs and the real-world networks they model. For example, Mihail, Papadimitriou, and Saberi [Mihail et al., 2006], as well as Gkantsidis, Mihail, and Saberi [Gkantsidis et al., 2003], studied Internet routing at the level of Autonomous Systems (AS), and showed that the preferential attachment model and a random graph model with power law degree distributions each have good expansion properties if the minimum degree is greater than 2 or 3, respectively. This is consistent with the empirical results, but as we have seen the AS graphs are quite unusual, when compared with nearly every other social and information network we have studied. On the other hand, Estrada has made the observation that although certain communication, information, and biological networks have good expansion properties, social networks do not [Estrada, 2006]. This is interpreted as evidence that such social networks have good small highly-cohesive groups, a property which is not attributed to the biological networks that were considered. From the perspective of our analysis, these results are interesting since it is likely that these small highly-cohesive groups correspond to sets near the global minimum of the network community profile plot. Reproducing deep cuts was also a

motivation for the development of the geometric preferential attachment models of Flaxman, Frieze, and Vera [Flaxman et al., 2004, 2007]. Note, however, that the deep cuts they obtain arise from the underlying geometry of the model and thus are nearly bisections.

Consider also recent results on the structural and spectral properties of very sparse random graphs. Recall that the G_{np} random graph model [Bollobás, 1985] consists of those graphs on n nodes, in which there is an edge between every pair vertices with a probability p , independently. Recall also that if $p \in (1/n, \log n/n)$, then a typical graph in G_{np} has a giant component, *i.e.*, connected subgraph consisting of a constant fraction of the nodes, but the graph is not fully connected [Bollobás, 1985]. (If $p < 1/n$, the a typical graph is disconnected and there does not exist a giant component, while if $p > \log n/n$, then a typical graph is fully connected.) As noted, *e.g.*, by Feige and Ofek [Feige and Ofek, 2005], this latter regime is particularly difficult to analyze since with fairly high probability there exist vertices with degrees that are much larger than their expected degree. As reviewed in Section 10.6.2, however, this regime is not unlike that in a power law random graph in which the power law exponent $\beta \in (2, 3)$ [Chung and Lu, 2001, Lu, 2001, Chung and Lu, 2006a].

Chakrabarti *et al.* [Chakrabarti et al., 2007a] defined the “min-cut” plot which has similarities with our NCP plot. They used a different approach in which a network was recursively bisected and then the quality of the obtained clusters was plotted against as a function of size; and the “min-cut” plots were only used as yet-another statistic to test when assessing how realistic are synthetically generated graphs. Note, however, that the “min-cut” plots have qualitatively similar behavior to our NCP plots, *i.e.*, they initially decrease, reach a minimum, and then increase.

Of particular interest to us are recent results on the mixing time of random walks in this $p \in (1/n, \log n/n)$ regime of the G_{np} (and the related G_{nm}) random graph model. Benjamini *et al.* [Benjamini et al., 2006] and Fountoulakis and Reed [Fountoulakis and Reed, 2007b,a] have established rapid mixing results by proving structural results about these very sparse graphs. In particular, they proved that these graphs may be viewed as a “core” expander subgraph, whose deletion leaves a large number of “decorations,” *i.e.*, small components such that a bounded number are attached to any vertex in the core. The particular constructions in their proofs is complicated, but they have a similar flavor to the core-and-whiskers structure we have empirically observed. Similar results were observed by Fernholz and Ramachandran [Fernholz and Ramachandran, 2007], whose analysis separately considered the 2-core of these graphs and then the residual pieces. They show that a typical longest shortest path between two vertices u and v consists of a path of length $O(\log n)$ from u to the 2-core, then a path of length $O(\log n)$ across the 2-core, and finally a path of length $O(\log n)$ from the 2-core to v . Again, this is reminiscent of the core-and-whiskers properties we have observed. In all these cases, the structure is very different than traditional expanders [Hoory et al., 2006], which we also empirically observe. Eigenvalues of power law graphs have also been studied by Mihail and Papadimitriou [Mihail and Papadimitriou, 2002], Chung, Lu, Vu [Chung et al., 2003a,b, 2004], and Flaxman, Frieze, and Fenner [Flaxman et al., 2005].

10.8 Conclusion

We investigated statistical properties of community-like sets of nodes in large real-world social and information networks. We discovered that community structure in these networks is very different than what we expected from the experience with small networks and from what commonly-used models would suggest.

In particular, we defined a *network community profile plot (NCP plot)*, and we observed that good network communities exist only up to a size scale of ≈ 100 nodes. This agrees well with the observations of Dunbar. For size scales above ≈ 100 nodes, the NCP plot slopes upwards as the conductance score of the best possible set of nodes gets gradually worse and worse as those sets increase in size. Thus, if the world is modeled by a sparse “interaction graph” and if a density-based notion such as conductance is an appropriate measure of community quality, then the “best” possible “communities” in nearly every real-world network we examined gradually gets less and less community-like and instead gradually “blends in” with the rest of the network, as the purported communities steadily grow in size. Although this suggests that large networks have a *core-periphery* or *jellyfish* type of structure, where small “whiskers” connect themselves into a large dense intermingled network “core,” we also observed that the “core” itself has an analogous core-periphery structure.

None of the commonly-used network generation models, including preferential-attachment, copying, and hierarchical models, generates networks that even qualitatively reproduce this community structure property. We found, however, that a model in which edges are added recursively, via an iterative “forest fire” burning mechanism, produces remarkably good results. Our work opens several new questions about the structure of large social and information networks in general, and it has implications for the use of graph partitioning algorithms on real-world networks and for detecting communities in them.

Chapter 11

Web projections: Learning from contextual subgraphs of the web

Graphical relationships among web pages have been leveraged as sources of information in methods for ranking search results. To date, specific graphical properties have been used in these analyses. We introduce *web projections*, a methodology that generalizes prior efforts on exploiting graphical relationships of the web in several ways. With the approach, we create subgraphs by projecting sets of pages and domains onto the larger web graph, and then use machine learning to construct predictive models that consider graphical properties as evidence. We describe the method and present experiments that illustrate the construction of predictive models of search result quality and user query reformulation.

11.1 Introduction

Information retrieval methods have traditionally considered documents as independent. A key insight coming to the fore with the pursuit of effective web search is that inferences about relevance can be enhanced by considering the hyperlink relationships among documents [Kleinberg, 1999a, Page et al., 1998]. We present a methodology we refer to as *web projections* that centers on the creation and the use of graphical properties of subgraphs of the web. With the approach, we project a set of web pages of interest, such as the results generated by a search engine for queries, on the larger web graph to extract a subgraph, which we call the *web projection graph*. We then identify and exploit graph-centric properties of this subgraph for a variety of search-related tasks. The method can be viewed as a general approach of using context-sensitive collections of web pages to define and focus attention on relevant subsets of the web graph, and then using graph-theoretic features within this subgraph as input to statistical models that can provide predictions about content, relevance, and user behavior.

We highlight in this chapter the use of the subgraphs for analyzing search result quality and for predicting user behavior in reformulating queries. Specifically, we investigate the following questions:

- How do query search results project onto the underlying web graph?
- What can we say about the quality of a set of search results, given the graph-theoretical properties of their projection on the web graph?

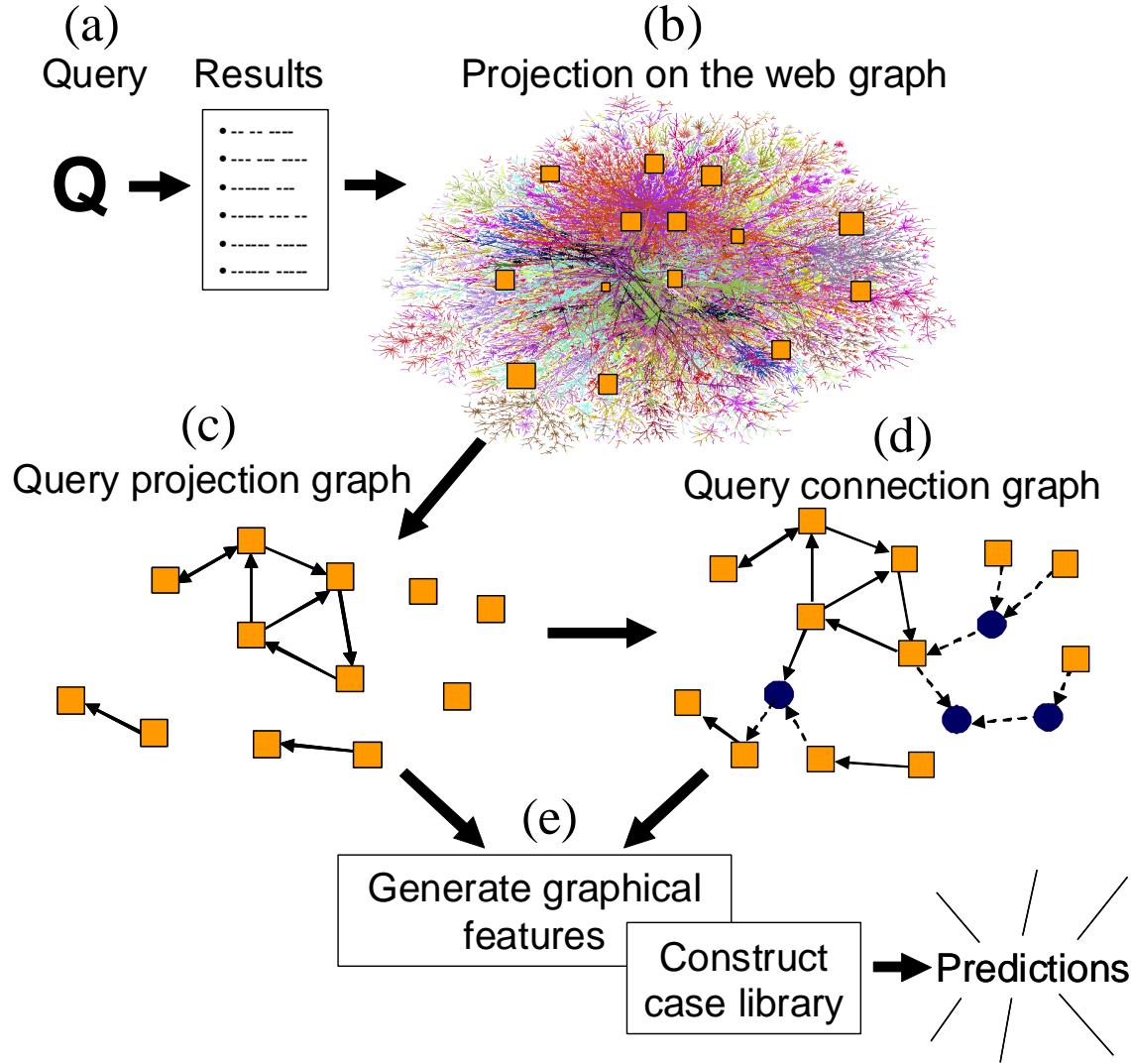


Figure 11.1: Web projection methodology. Given a query and respective search results, a query projection graph is generated and graph-theoretic features are then used for building predictive models.

- Can we predict the difficulty of the query given the projection graph?
- Can we predict users' behaviors with issuing and reformulating queries given the query projection graph?
- How do query reformulations reflect on the query projection graphs?

The rest of the chapter is organized as follows. In Section 11.2, we introduce web projections and explain the methodology and the attributes used to model query projection graphs. In Section 11.3, we describe the data used in our studies. We then describe applications of our approach to predict the quality of sets of search results (Section 11.4), and to model user behavior when reformulating queries (Section 11.5). In Section 11.6, we compare our work to prior research. Finally, we summarize and conclude in Section 11.7.

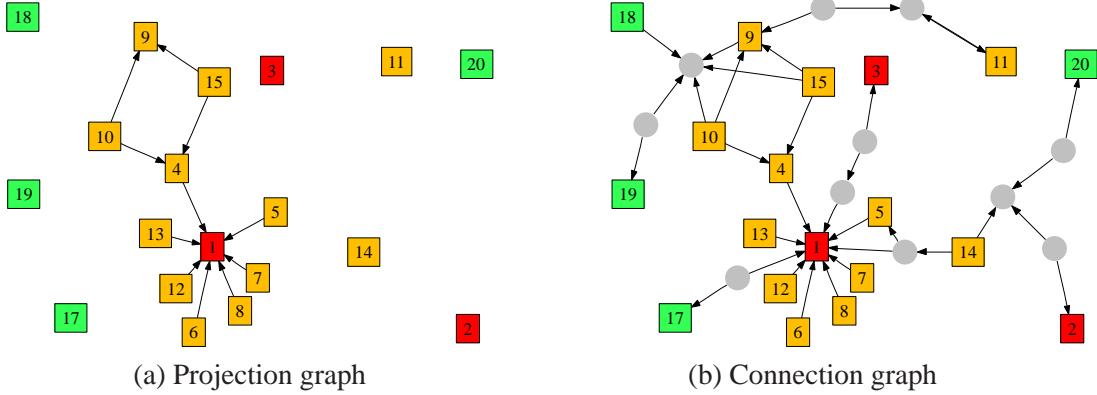


Figure 11.2: Query projection graph and query connection graph for the top 20 results of the query *Yahoo search engine* projected on the URL graph.

11.2 Query projections

We begin by describing the main steps with building web projections and then provide formal definitions of the key components. Figure 11.1 shows the basic steps of applying the method to analyze search results. We start with a query and a set of results for the query, generated by some procedure, typically via the use of a preexisting search engine (a). We project the search results on the web graph (b), by finding the search results (square nodes) in the larger web graph and then inducing a subgraph based on these identified nodes (c). We name the induced subgraph the *query projection graph*.

Given the typical distances among search results, the query projection graph often contains disconnected components. We connect the nodes of query projection graph to create a *query connection graph* (d). The disconnected components of the query projection graph are connected by identifying web pages that join the components via shortest paths. The *connection nodes* that are introduced during the connecting of the projection graph (circular nodes in Fig. 11.1) are not drawn from the search result set.

Given the query projection graph and query connection graph we generate a set of evidential features describing the topology of the graph for use in the creation of predictive models via machine learning (e). We provide details about sample topological features in Section 11.2.2. Finally, we build a case library from a consideration of the topological properties for multiple queries for different outcomes (e.g., high-quality versus low-quality sets of results), and use the case library of graph-theoretic relationships and outcomes to train models that can make predictions about the outcomes. We shall focus in this chapter on the tasks that harness graphical properties of web projections generated from sets of results. Two such tasks are the construction of statistical models for predicting quality of a set of search results and modeling user behavior in reformulating queries. As we shall discuss later, there are also opportunities to use the web projection approach to assist with the ranking of individual search results. In such applications, properties and relationships of single results to the subset of pages in the query projection are of interest.

Figure 11.2 shows an example of a query projection graph and query connection graph for the query *Yahoo search engine*. Square nodes represent web pages and directed edges represent hyperlinks. Circular nodes represent connection nodes. The number in each square represents the rank of the search result in the list returned by a search engine. The color (monochromatic shade) of the node indicates a human-evaluated relevance score of the result to the query. The most relevant results are colored dark (red), the next most

relevant orange, then yellow, green, blue and purple. Figure 11.3 shows the projection of results for the query *Subaru* onto the domain graph rather than the URL graph. For both projections, the most relevant nodes (colored dark) appear in central locations in the graph and are pointed at by other search results. In contrast, the least relevant nodes (colored bright) are usually not as well connected, often requiring connection nodes to join them to the subgraph.

11.2.1 Query projection and connection graphs

We now present formal definitions of query projection graph and query connection graph. Consider a *directed* web graph $G(N, E)$ with node set N and directed edge set E , and a given set of search results S . First, we project the results on the web graph G to obtain a set of *projection nodes* N_p , where $N_p = S \cap N$. Note that, ideally, we would like $N_p = S$ but since the coverage of the web graph may not be complete, some search results may not be found in the graph. Thus, in general, $N_p \subseteq S$. We define:

- **Query projection graph** is a subgraph $G_p(N_p, E_p)$ of G induced on N_p nodes, *i.e.*, edge set $E_p = \{(u, v) \in E; u \in N_p \wedge v \in N_p\}$
- **Query connection graph** is a subgraph $G_c(N_c, E_c)$ of G induced on N_c nodes, where $N_c = N_p \cup C$, *i.e.*, edge set $E_c = \{(u, v) \in E; u \in N_c \wedge v \in N_c\}$. Set C is a set of connection nodes, *i.e.*, minimal set of nodes that makes graph G_p connected.

Note that finding the minimal set of connection nodes C is NP-hard, since the problem of finding a Steiner tree [Karp, 1972] reduces to this problem. In our experiments, we used a heuristic policy to find the set C , *i.e.*, to connect the components of G_p . We found that the heuristic policy was reliable and performed well on the datasets that we considered. The policy is as follows:

Let D_i denote the node sets of connected components of G_p ordered by decreasing size ($|D_i| \geq |D_{i+1}|$). We connect each component via the shortest path to the largest component and continue until all components are connected. More precisely, we start with D_2 and connect it via a shortest path on nodes C_2 to D_1 . This creates a new largest component $D_{12} = D_1 \cup C_2 \cup D_2$. Now, we proceed and connect D_3 to D_{12} via shortest path C_3 , creating $D_{123} = D_{12} \cup C_3 \cup D_3$, and so on until all components are connected. A set of *connection nodes* C is then $C = \bigcup C_i$. We define a shortest path between the sets of nodes U and V as the *shortest undirected path* over all pairs of nodes (u, v) , $u \in U, v \in V$.

11.2.2 From graph to evidential features

Given query projection and connection graphs, we seek to extract a set of features that captures key properties of the topology of the graphs. In all, we considered 55 features to describe the characteristics of the projection and connection graphs, and of the query.

Table 11.1 presents representative features drawn from the larger set of attributes that we used in our experiments. The majority of the features are graphical properties, including the number of nodes and edges, the number and size of connected subgraphs, etc. See [Wasserman and Faust, 1994] for a review of basic graph-theoretic concepts and detailed definitions of the features we use in this work. In one set of experiments, we also considered non-topological properties derived solely from the text of the query and results.

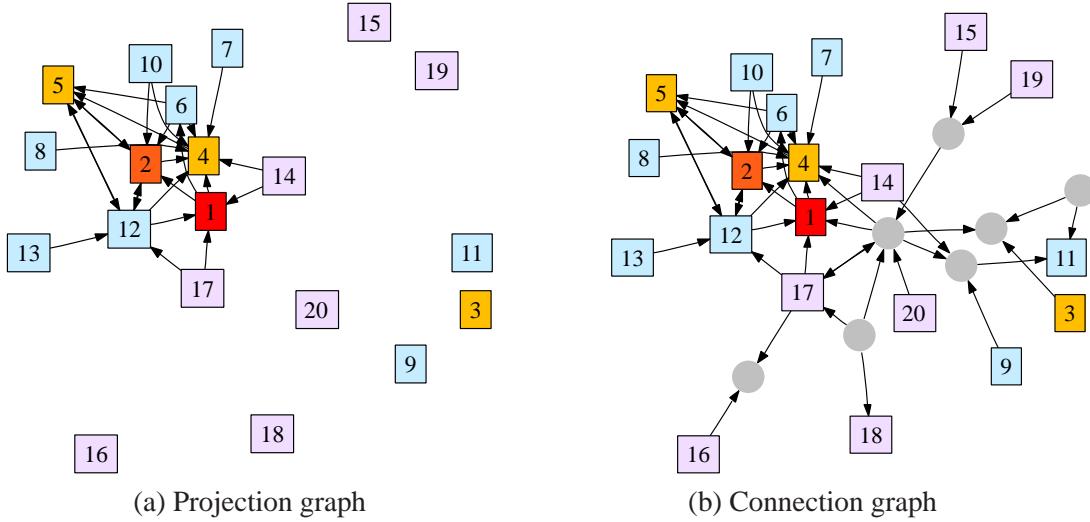


Figure 11.3: Query projection graph and query connection graph for the top 20 results of the query *Subaru* projected on the domain graph. Notice that projection on the domain graph is denser than projection on the URL graph (figure 11.2).

We group the evidential features into four classes: *Query projection graph features* (GF-PROJ, 12 features) are calculated from the projection graph. These features measure various aspects of the connectivity of G_p . Similarly, *query connection graph features* (GF-CONN, 16 features) are obtained from G_c and aim to capture the relations between the projection nodes N_p in the context of connection nodes C . We also consider *combination features* (GF-COMB, 17 features), defined as compositions of features from the other groups. They largely include various ratios and normalizations of more atomic features contained in the other categories. Last, *Query features* (F-QUERY, 10 features) represent non-graphical properties of the result set, calculated from the text of the query and a list of returned search results, including the number of results and domains in the result set.

11.3 Generating case libraries

We now present details on constructing libraries of cases consisting of sets of topological properties that characterize the projections of queries onto the web graph. We used two different representations of the web as a graph. For the study of relevance, we constructed projections from nearly 30 thousand queries, each with a corresponding set of search results. Most of the search results were labeled with a human-assigned relevancy score. For our study of query reformulations, we employed a set of 42 million query-to-query transitions with corresponding lists of search results generated at each step in the search session. For all of the experiments, the search results were obtained from a state-of-the-art ranking algorithm which considers a large number of content features as well as some topological properties on the web as a whole.

GF-PROJ: query projection graph (G_p) features (12)	
G_p Nodes	number of nodes in G_p
G_p Edges	number of edges in G_p
G_p Components	number of connected components
G_p LccNodes	nodes in largest component
G_p LccEdges	edges in largest component
G_p MxDeg	maximal node degree
G_p Deg0Nodes	number of isolated nodes
G_p Deg1Nodes	number of degree 1 nodes
G_p Triads	number of triangles in G_p
G_p Density	density of G_p ($ E_p /(N_p (N_p - 1))$)
G_p LccSize	size of largest component ($ D_1 / N_p $)
G_p Clustering	clustering coefficient of G_p
GF-CONN: query connection graph (G_c) features (16)	
G_c Nodes	number of nodes in G_c
G_c Edges	number of edges G_c
G_c CNodes	number of connector nodes C
G_c CEdges	number of edges incident to C
G_c MxCnDeg	maximal connector node C degree
G_c MxCnOutDeg	maximal connector node C out-degree
G_c MxPnDeg	max projection node (N_p) degree in G_c
G_c AvgPnPath	mean path length of N_p nodes in G_c
G_c MxPnPath	max path length of N_p nodes in G_c
G_c AvgPath	mean path length of N_c nodes in G_c
G_c MxPath	max path length of N_c nodes in G_c
G_c Triads	number of triangles in G_c
G_c Density	density of G_c ($ E_c /(N_c (N_c - 1))$)
G_c Clustering	clustering coefficient of G_c
GF-COMB: Combined features (17 features)	
DomsToUrls	Ratio of domains to urls in result set
Coverage	Coverage of the projection (N_p/S)
G_pG_c Nodes	Node ratio ($ N_p / N_c $)
G_pG_c Edges	Edge ratio ($ E_p / E_c $)
G_pG_c AvgPath	Path ratio (AvgPnPath/ G_c AvgPath)
G_pG_c MxPath	Path ratio (MxPnPath/ G_c MxPath)
F-QUERY: query features (10 features)	
QueryChLen	number of characters in the query
QueryWrdLen	number of query words
QuerySrcRes	number of search results
QueryNDoms	number of domains in result set
QueryNUrl	number of URLs in result set
QueryNRated	number of results with human rating

Table 11.1: Sample features used to represent query projection, and connection graphs, and the query.

11.3.1 Web as a graph

We now present the web graphs that provided the substrate for the query-focused projections. We use two variants of the web graph: a URL graph and a domain graph.

URL graph

URL graphs are the most commonly used representation of the web as a directed graph. Nodes represent web pages, and there is a directed edge from node u to node v if there is a hyperlink from web pages u to web pages v . We created our web graph based on a sample of 22 million web pages from a crawl of the web performed in March 2006. We used a sample of the web considered to be of high quality. We started crawling from a seed set of popular, high quality web pages with good reputation. The graph contains 345 million edges and is well connected; the largest weakly connected component contains 21 million nodes, while the second largest has less than a thousand nodes. The strongly connected component is also large, containing 14 million nodes. The second largest component has 7000 nodes. The graph has diameter of 8, and node degrees follow a power law distribution.

For some prediction tasks, we focused on subsets of these URLs, e.g., those for which we have human relevance judgments. When we project the URLs tagged with relevance judgments onto this URL graph, results may be missing. URLs may be absent for several reasons, including the limited nature of the sample of URLs that we worked with, changes in pages that are returned and judged over time, and the volatile nature of dynamically generated pages. For some tasks, (e.g., predicting the top 20 versus bottom 20 results set, described in detail in Section 11.4.3), the difference in coverage alone can be a good predictor of class. As we wanted to focus more on the graphical properties than on coverage per se, we normalized the number of projected results in the graph for the different classes. We did this by first noting how many URLs for the top 20 results were in the projection graph, then considering as many results as needed from the bottom to get the same coverage.

Domain graph

In the domain graph, nodes represent domain names, (e.g., `cmu.edu` or `microsoft.com`), and there is a directed edge from node u to node v , if there are web pages inside domain u that contain a hyperlink to web pages inside domain v . It is important to note that nodes in a domain graph are not arbitrary domains; all sub-domains are collapsed into a second-level domain name. For example, web pages from domains `cs.cmu.edu`, `m1.cmu.edu`, and `lti.cs.cmu.edu` are merged into a single node (domain name) `cmu.edu`.

We considered a complete domain graph of the web from February 2006. The graph contains 39 million domain names and 720 million directed edges. The graph is densely connected, has a diameter of 4, and the largest component contains 99.9% of the nodes. Since this is a complete domain graph we have no problems with projection coverage. The domain of every search result in our dataset can be found in this graph.

Figure 11.4 shows the differences between projections on URL and the domain graph when projecting the top 20 results for the query *encyclopedia*. Domain graph projections are usually denser and much better connected than the URL graph projections. Domain graphs also have better coverage of the search results, with very few missing nodes.

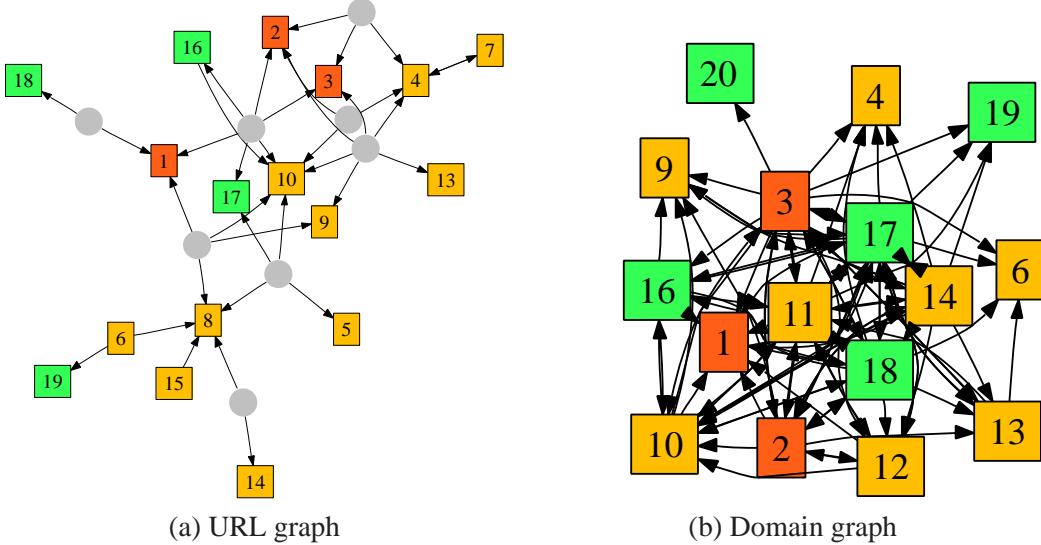


Figure 11.4: Query projection graph for the top 20 results of the query *encyclopedia* projected on the URL and domain graphs.

11.3.2 Human-rated search results

In one set of experiments, we explored the use of the web-projection methodology for the task of predicting the quality of a set of search results. This task requires assessments of result quality, which we obtained from human judges. For each query, the top k results from one or more systems were presented to the judges for evaluation. The quality of a query-result pair was explicitly labeled by the judges using a six point scale ranging from “Perfect” to “Bad”. We note that the labeling was performed over the results already highly ranked by a web search engine, and thus corresponds to a typical user experience. Out of 30,000 total available queries, we focused our experiments on a set of 13,000 queries with at least 40 rated results, with averages of 57 results (URLs) and 46 domains per query.

11.3.3 Query reformulation corpus

In a second set of experiments, we used web projections to explore patterns of query reformulation. We examined a sample of query logs captured over a six week period by a popular web search engine. We obtained query-query transitions from the logs as described in [Radlinski and Dumais, 2006]. For each query q_i , we measured n_i , the number of times the query was observed. For a pair of queries (q_i, q_j) , we also measured the probability of reformulation or transition from query i to j , p_{ij} . If we let n_{ij} be the number of times that q_i was followed by q_j within a thirty-minute window, then $p_{ij} = n_{ij}/n_i$ is the probability of q_i being followed by q_j . And similarly, probability p_i of query q_i participating in a transition is defined as $p_i = \sum_j n_{ij}/n_i$.

We started with a set of 35 million queries and 80 million query transitions as defined above. For the analysis described below, we considered only queries and reformulations that appeared at least 10 times in our corpus. Our analyses used 48,458 queries and 120,914 query transitions. We then used the top 20 search results for each of the 48 thousand queries and projected them on the URL and the domain graphs.

11.4 Quality of search results

For predicting the quality of search results, we asked the following questions: By analyzing the projection of a query onto the web graph, what can we tell about the quality of the returned result set? What can we tell about the difficulty of the query? More specifically, we explored the following two tasks:

1. Discriminate good (top 20) versus poor (bottom 20) search result sets.
2. Given a set of results, predict how good the set is, *i.e.*, predict the highest human relevancy rating in the set.

We now describe the problem setting and experimental setup as well as the baseline method.

11.4.1 Problem definition

We focus on the following general setting: We are given a query q_i with a set of search results S_i . Each query q_i belongs to class y_i . A class is a categorical value that can be, as an example, the rating of the most relevant search result in the set, or an indicator of whether the result set S_i is composed of the top-ranked or bottom-ranked search results.

We start with S_i and project it on both the URL and the domain graphs (see Section 11.3.1), create both projection and connection graphs, and extract the attributes as described in Section 11.2.2. This means that we project every query q_i onto two different graphs of the web, and for each projection, we extract a set of features as defined in table 11.1. We generate a case library of training examples q_i described with features and we learn a model to predict class y_i via a machine learning procedure.

11.4.2 Experimental setup

For learning the models, we used the WinMine toolkit [Chickering, 2002] that uses the GES [Chickering, 2003] algorithm in Bayesian structure search to learn a Bayesian network. We model the continuous features as Gaussians and discrete features with a multinomial distribution. For all experiments we report the average classification accuracy over a 10-fold cross validation.

We compare the predictive power of the learned models with two baseline methods. The first baseline model is the marginal model, which predicts the most common class. The second baseline algorithm we use is based on a ranking algorithm that uses a large number of textual and global graph features to rank search results. For the classification tasks, we learn a threshold on the score to predict the class.

The baseline ranking algorithm we used is RankNet [Burges et al., 2005], a supervised machine-learning technique developed to learn a ranking function. The learning methodology is a neural net algorithm that optimizes feature weights to best match explicitly provided pairwise user preferences. Over 350 input features are used to train RankNet. These features include various aspects of document content, anchor text features, and basic hyperlink features. The output of RankNet is used to rank results. Since the output of RankNet is a combination of state of the art features for ranking, we use it as a discriminatory feature that serves as input to the Bayesian-network classifier. We think this serves as a strong baseline.

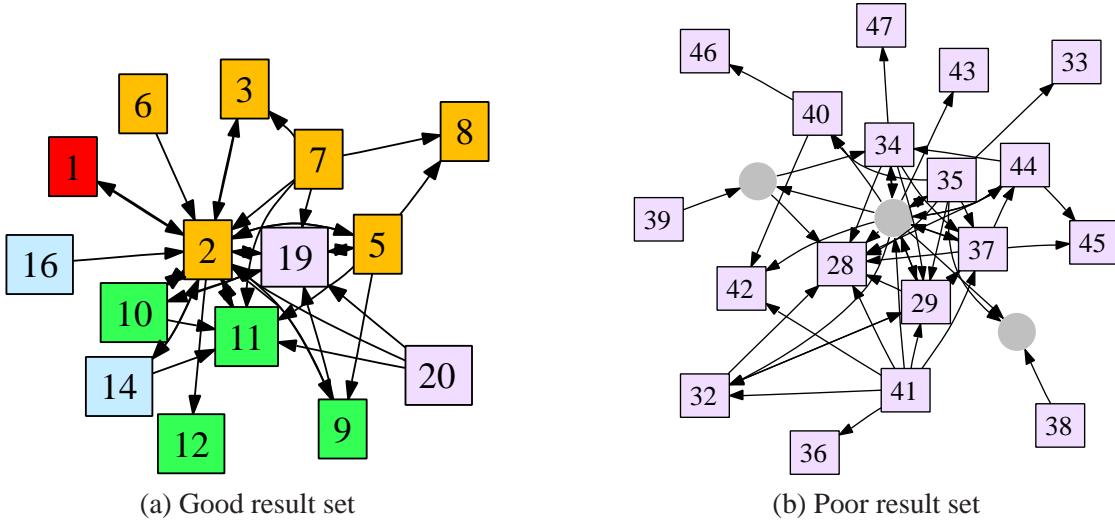


Figure 11.5: Domain graph projections of good and poor result sets for query *medline*.

11.4.3 Relative quality of result sets

The first task we consider is the classification of good (top 20) versus poor (bottom 20) result sets. For this task, we used the explicit relevance judgments described in 11.3.2. For each query, we order the search results from best to worst using the human judgments. We note that this ordering can be different than the output of the search engine.

We then project the top 20 search results, and bottom 20 results ordered by human judgments onto the URL and domain graphs, and compute the features described in table 11.1 for the two graphs. We learn a model that can predict, for a previously unseen query with a set of search results, whether it is good (top 20) or poor (bottom 20). Given the average number of judged search results per query, we are effectively learning to discriminate the top 20 results versus the search results with ranks 40 to 60. Note that this task involves predicting the relative quality of results for a given query. We examine predicting the absolute quality of the results in the next section.

First, we point to Figure 11.5 which displays examples of projections of good and poor result sets for the query *medline* on the domain graph. We can see that results in a good set are tightly clustered, while those in the poor result set are more spread out, requiring many connection nodes to connect the components of the projection graph. Also notice that the results marked by humans as most relevant (darkest nodes) appear as the central (high-degree) nodes in the graph (Figure 11.5(a)). Similarly, Figure 11.6 shows the good and poor result sets for the query *Wisconsin* projected on the URL graph. The central node in the good result set (panel a) is one of the search results, whereas in poor set (panel b) the central node is a derived connector node.

Table 11.2 shows the results for the task of predicting good versus poor result sets using several different methods and feature sets. Results are shown separately for URL and domain graph projections.

The Baseline–Marginals row displays the classification accuracy of predicting the most common class. Baseline–RankNet is the second baseline where only the RankNet score is used for learning. We are using a combination of about 350 textual features to discriminate the good and the poor result sets. GF-PROJ

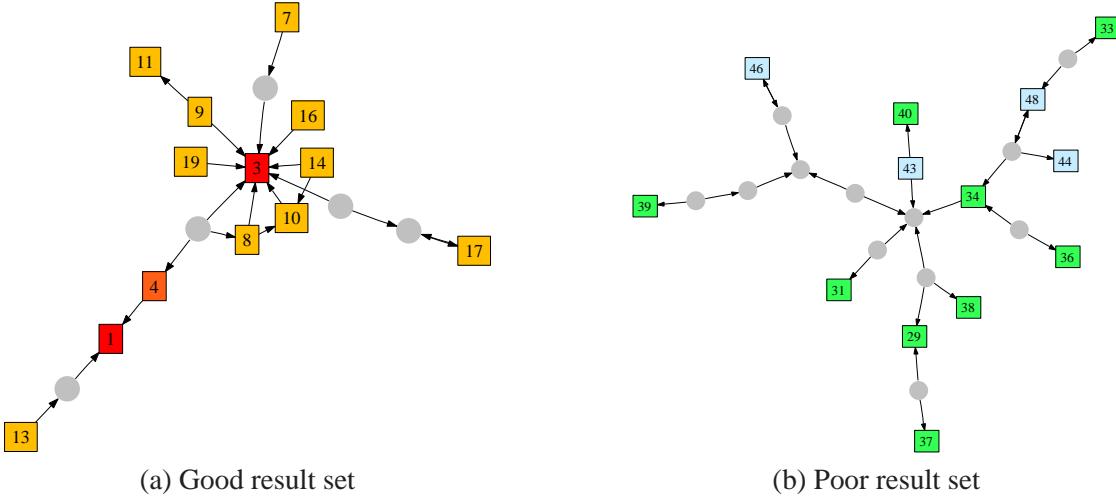


Figure 11.6: Projections of good and poor result sets for query *Wisconsin* projected on the URL graph.

Feature set	URL graph	Domain graph
Baseline–Marginals	0.50	0.50
Baseline–RankNet	0.74	0.74
GF-PROJ	0.62	0.82
GF-CONN	0.60	0.86
GF-PROJ+GF-CONN	0.87	0.90
GF-ALL	0.88	0.88

Table 11.2: Classification accuracy for predicting good versus poor result sets.

uses the 12 features extracted from the projection graph, GF-CONN uses the 16 connection graph features, GF-PROJ+GF-CONN uses both of these feature sets (28 features), and GF-ALL refers the case where all 55 features, most of which are described in table 11.1, are used for learning.

We were not surprised to find that RankNet and the new graphical features outperformed the marginal baseline. The RankNet output reflects the extent to which human judgments agree with the output of the learned ranking function. The “GF-” results reflect the extent to which graphical features of the results subset are predictive of human relevance judgments. For the URL graph, the RankNet baseline outperforms models trained only on projection or connection graph features, but the models trained on both sets of features shows substantial improvement over RankNet (18% relative improvement). For the domain graph, all models trained on graphical features outperform RankNet. We obtained the best classification accuracy of 90% when combining projection and connection graph features. We note that we obtained higher classification accuracies when projecting on the domain graph than for URL projections. This is likely due to the sparser coverage of the URL graph.

We found interesting the performance of the “GF-” models, considering only topological features of the web projections, and bypassing analysis of content matches between the query and web pages.

Figure 11.7 shows a simple model learned from the query projections on the domain graph using the GF-PROJ feature set. The model has a classification accuracy of 0.82. The figure shows the decision tree for the output variable Top 20 vs. Bottom 20. Nodes correspond to input features, and each leaf node

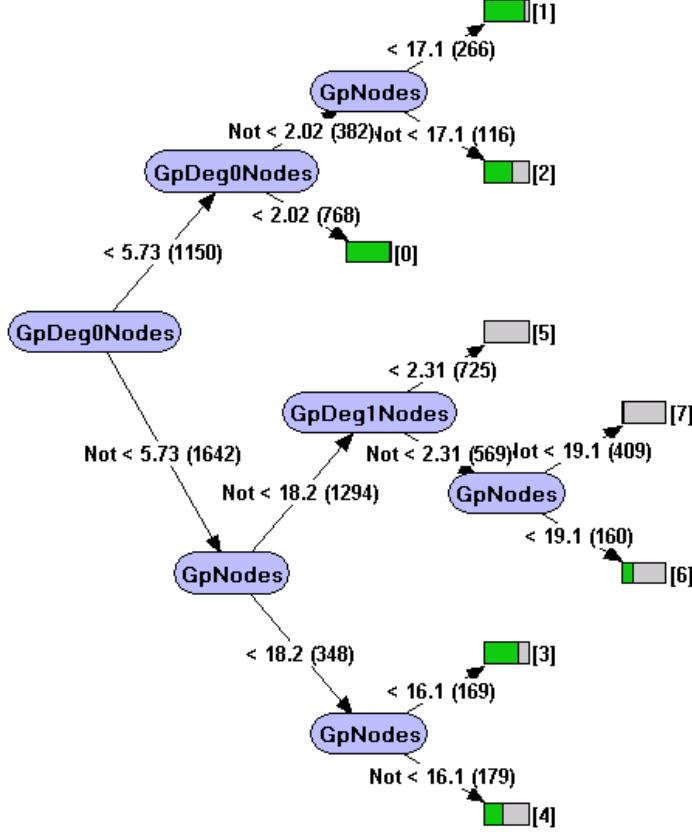


Figure 11.7: Learned model for discriminating good versus poor search result sets based on query projection on the domain graph.

shows the probability distribution for the output variable, which is shown as a histogram. In this case, the variable has only two possible values; the green (darker) area indicates the proportion of good (top 20) sets and the grey area poor (bottom 20) sets. Labels on the edges show the splitting criteria of the parent node variable, and the numbers in parenthesis show the number of training examples routed over the edge. The projection graphs of good result sets (shown as large green (dark) area of the histograms) have few isolated nodes (low values of G_p Deg0Nodes) and results are coming from a few domains (low values of G_p Nodes). On the other hand, poor result sets have many domains and many isolated nodes.

11.4.4 Absolute quality of a result set

In the previous section, we considered the problem of discriminating between good and poor result sets. Now we focus only on top-rated (top 20) results for each query and aim to predict the absolute quality of a query result set. More specifically, we label each query with the highest human-assigned rating for any result in the set. We note that we could use other measures to summarize the quality of result sets. The highest human rating is easy to describe and is of practical importance. Since the human relevance judgments were on a 6-point scale (Section 11.3.2), we can examine the problem at several granularities. Here, we present results for the 6-class problem (predict top label exactly) and the 2-class problem (predict whether the top label is from the three highest or three lowest rating categories).

Feature set	URL graph	Domain graph
Baseline–Marginals	0.36	0.36
Baseline–RankNet	0.48	0.44
GF-PROJ	0.51	0.53
GF-CONN	0.50	0.52
GF-PROJ+GF-CONN	0.54	0.54
GF-ALL	0.55	0.55

Table 11.3: Result set quality classification accuracy for a 6-way classification problem.

Feature set	URL graph	Domain graph
Baseline–Marginals	0.55	0.55
Baseline–RankNet	0.63	0.60
GF-PROJ	0.80	0.64
GF-CONN	0.79	0.66
GF-PROJ+GF-CONN	0.82	0.69
GF-ALL	0.83	0.71

Table 11.4: Result set quality classification accuracy for a binary classification problem.

First, we consider the 6-class task of predicting the exact rating of the highest rated document in the set of top 20 results. Table 11.3 shows the classification results. For the URL graph, we obtained a 15% relative improvement over using the RankNet to predict the quality when using all attributes. For the domain graph the improvement was even larger, 25%. Note that all methods using any combination of graphical attributes outperform both baseline methods.

The model (not displayed per space limitations) for the 6-level result set quality classification problem is more complex. The first split of the induced decision tree is on the node ratio of the projection and connection graphs. If the connection graph is much larger than the projection graph, the results are likely to be of poor quality. Moving down the tree, we see that if maximum degree in a graph is relatively small, the results are likely to be of medium quality, with results getting worse as the number of domains in a top 20 set increases. The model revealed that high quality search result sets are associated with projection nodes with large degrees, few domains, small domains to URL ratios, and are well connected.

Next, we examine the same problem at a coarser granularity. The task is to predict whether the set contains a result with the rating in the top or the bottom half of the 6 point rating scale. Table 11.4 shows the classification accuracies for the classification problem. We note that the difference in performance between the domain and URL graph projections increased even further and that the relative increase in performance over the RankNet baseline increased (31% for the URL and 18% for the domain graph).

This task is similar to that of discriminating the good versus poor result sets (as described in Section 11.4.3). However, it is also more difficult since we are only working with top 20 results for each query and predicting the absolute quality of the set. The good versus poor prediction requires only a relative judgment.

For the task of distinguishing good versus poor result sets, we found that projections on the domain graph outperformed the projections on the URL graph. For the case of predicting the exact quality of a result set, the projections on the URL graph generally performed better even in cases where the URL graph has

the problems with coverage. This may be explained by the difference in the goals and representation. For good versus poor discriminations, the quality of the whole set is important and the domain graph likely represents an appropriate level of abstraction for handling this challenge. In contrast, the quality of a result set is a single result (single node) property. Here the domain graph may be too coarse to capture fine-grained properties of the high quality nodes (search results). A projection on the URL graph may be needed to capture necessary properties.

11.5 Query reformulations

As a second illustration of the use of web projections, we explore the learning of models to predict users' query-reformulation behavior and characteristics. Web searchers often refine their queries one or more times, as they seek information on the web. Prior research has explored query reformulations, considering such issues as the timing and type of reformulation seen. For example, Lau and Horvitz [Lau and Horvitz, 1999] build models to predict the likelihood that searchers will specialize, generalize, or reformulate queries within a search session, considering the history and timing of actions. Jones *et al.* [Jones et al., 2006] examine substitutions that searchers make to their queries.

We explore the use of web projections to build models that predict if and how users reformulate their queries. We used a set of 48 thousand queries that were reformulated at least 10 times. For every query, we took the top 20 search results returned by the search engine, and created the query projection and connection graphs, extracted the graph features, and trained predictive models.

More specifically, we consider the following tasks:

1. Distinguish queries with high versus low reformulation probability.
2. Given a transition from query q_s to query q_d , predict whether it is a specialization or generalization.
3. Given a query that is likely to be reformulated, predict whether it is going to be generalized or specialized.

Next, we describe the experimental setup and give more detailed description of our results and findings.

11.5.1 Experimental setup

Using the query reformulation data described in Section 11.3.3, we defined several binary classification tasks. For each selected query, we took the top 20 search results as returned by the search engine, projected them on the domain and URL graphs, and extracted the features. For some of the tasks, the training datasets were quite imbalanced where one outcome was significantly more likely than the other. In order to focus on the key discriminations rather than basic marginal frequencies, we sub-sampled the majority class, so that both classes had roughly the same number of training examples.

11.5.2 Probability of query reformulation

First, we considered the problem of learning whether a query is likely to be reformulated or not. We split our set of queries into two classes: queries with high reformulation probability ($p_i \geq 0.6$) and queries with

Feature set	URL graph	Domain graph
Baseline–Marginals	0.54	0.56
GF-PROJ	0.59	0.58
GF-CONN	0.63	0.59
GF-PROJ+GF-CONN	0.63	0.60
GF-ALL	0.71	0.67

Table 11.5: Classification accuracy of predicting whether the query is likely to be reformulated.

low reformulation probability ($p_i \leq 0.15$). We selected these values so that the two classes were about the same size.

Table 11.5 shows the classification accuracies when projecting on URL and domain graphs. We found gradual improvement with increasing the numbers of topological features under consideration. We also found that cases drawn from projections on the URL graph provided better performance than cases generated from projections on the domain graph. We note the baselines for predicting the most common class are slightly different between the URL and the domain graph since we discarded a few queries that produced very small URL projection graphs.

Examining the model (not shown as the model is too large) we see that, queries that are likely to get reformulated come from many domains, are generally longer than queries that are not reformulated, and have relatively high degree (> 4) connection nodes. Such findings again suggest that queries whose results are tightly knit together on the web are of higher quality, given that such queries are less likely to be reformulated. The findings also suggest result sets with central high degree nodes and a small number of connector nodes are of higher quality.

In another set of experiments, we explored transitions between queries. For this task, we took pairs of queries where there is a strong tendency of transition in only one direction, and then trained a model that learns whether a given query is likely to be the transition *source* or *destination*. Figure 11.8 shows two examples of source and destination graphs. Our models were able to predict whether a given query is a source or a destination of the transition with an 85% classification accuracy. The learned model provided insights about the relationship between topological properties and the likelihood of the direction of a transition. We saw that sources of query transitions tend to have some isolated nodes, short query strings, many connected components, and nodes that lie far apart in the connection graph, which indicates the returned search results are not satisfactory. In contrast, reformulation destinations (especially specializations) tend to be better connected, and to have higher in-degrees of projection nodes. Intuitively, these results make sense: a searcher probably wants to specify a new query if the search results are somewhat “random”, *i.e.*, are scattered widely around on the web. The results of this experiment led another question, which we explore in the following section.

11.5.3 Query specialization versus generalization

We have just described how we can reliably learn whether a given query is the source or destination of a reformulation. Now, we pursue models that can predict the nature of reformulations. We shall explore in particular whether a reformulation is likely to be a specialization or a generalization of the source query. Given a pair of queries, where q_s is often reformulated into q_d , we want to learn characteristics of projection graphs for queries that are specialized versus generalized.

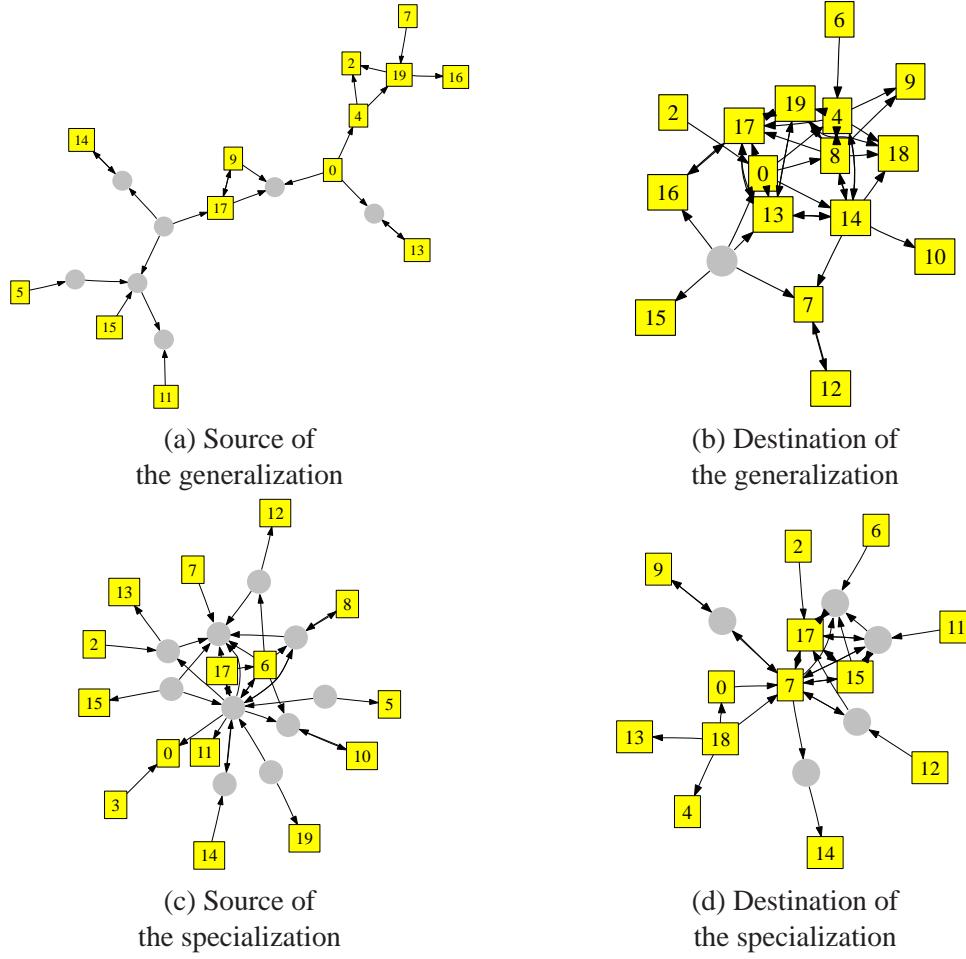


Figure 11.8: Sources and destinations of query transitions. Projections (a) and (b) show an example of a generalization from query *free house plans* to the query *house plans*. Projections (c) and (d) show the specialization from *strawberry shortcake* to *strawberry shortcake pictures*. Notice how the reformulated queries result in more connected graphs and bring result nodes into the center.

For this task, we define query specialization as the addition of more words to an existing query, and similarly define generalization as removing words from the query (richer characterizations have been considered Lau and Horvitz [Lau and Horvitz, 1999] and by Jones *et al.* [Jones *et al.*, 2006].) Given the query transition data, we extracted all pairs of queries where a specialization or generalization transition had occurred at least 10 times. Then we separately projected the source q_s and the destination query q_d and extracted the features. We created transition features by simply taking the difference of the corresponding feature values: $F_i(q_s) - F_i(q_d)$, where $F_i()$ denotes i^{th} feature. Note that in this experiment we do not use the query text attributes (length of the query string) as it would be possible to directly identify the type of transition by the change in the length of the query.

We show the classification performance in Table 11.6. Here, using only the projection graph features performs slightly better than using solely the connection graph features. We see consistent increases in performance by combining the projection graph and derived features. We obtain the best accuracy of 87% using projections on the domain graph and all features for learning the predictive models.

Feature set	URL graph	Domain graph
Baseline–Marginals	0.50	0.50
GF-PROJ	0.71	0.84
GF-CONN	0.69	0.83
GF-PROJ+GF-CONN	0.71	0.85
GF-ALL	0.80	0.87

Table 11.6: Classification accuracy of predicting whether a given query transition is a specialization or a generalization.

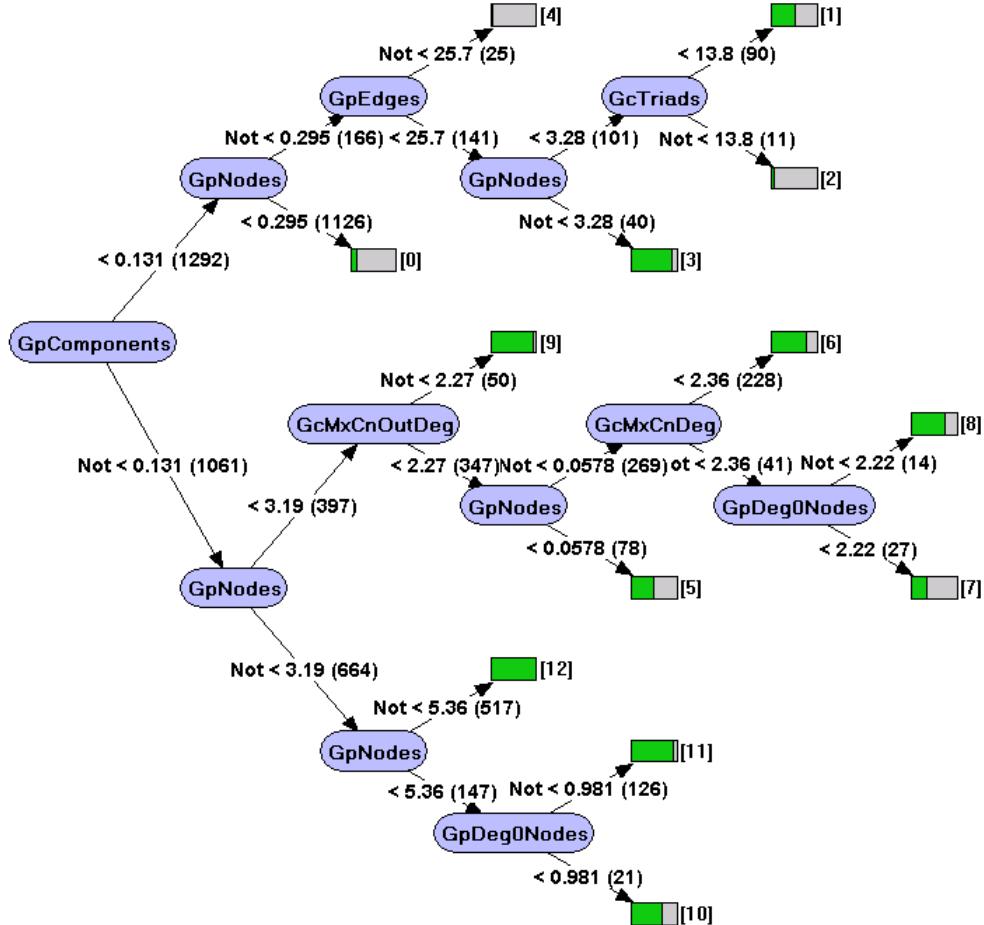


Figure 11.9: Model learned on URL graph projections for predicting whether transitions between queries are generalizations or specializations.

The learned decision tree for predicting reformulation using GF-PROJ+GF-CONN features with projections on the URL graph is displayed in Figure 11.9. Note that the splitting criteria (*e.g.*, GpComponents) here are not the values of the attributes but rather the changes in attribute values, *i.e.*, the difference in the attribute values of the source and the destination of the transition. The model shows that query specializations are characterized by the decrease in the number of connected components of projection graph (first split of the tree). It also shows that the number of nodes and edges in the projection graph increases

Feature set	URL graph	Domain graph
Baseline–Marginals	0.50	0.50
GF-PROJ	0.71	0.68
GF-CONN	0.62	0.65
GF-PROJ+GF-CONN	0.70	0.68
GF-ALL	0.78	0.76

Table 11.7: Classification accuracy of predicting whether a reformulation will likely lead to a specialization or generalization.

for generalizations. For specializations, we see that the number of isolated nodes decreases, results are gathered in few connected components, and the size of largest connected component is increases, while the number of connector nodes decreases. These results correspond with the intuition that, when a query is generalized, the list of results will get richer and more diverse. The findings revealed in the learned models suggest that the projection graphs associated with generalizations are sparser and less connected than those associated with specializations, where the projection is likely to be more concentrated, denser, requiring fewer connector nodes.

It may seem that the results here do not go along with those in section 11.5.2, where we find characteristics of query projection graphs that lead to query reformulation, *i.e.* learn characteristics of badly formulated queries and transitions as query is formulated. On contrary, we see here that in general specializations narrow down the search, while generalizations tend to lead to higher diversity and larger coverage.

11.5.4 Predicting type of query reformulation

Finally, we examine the type of reformulation associated with a query. We seek to predict whether it is more likely to see specific queries generalized or specialized, and how this reflects on the properties of the query projections. For this task, we learn models that consider specific properties of queries that are reformulated in a certain way. Again, we do not use the features derived from the query string (length of the query, number of words, etc.) as the change in the length of the query provides information about the type of reformulation.

Table 11.7 gives the classification performance for these models. We found that the performance of models learned from cases generated from the URL and domain graph projections is about the same. The projection graphs provided models with better performance than those using only the features of connection graph. Using all features, we obtained a classification performance of 78%.

The learned probabilistic decision tree model shows that the most discriminatory property for this task is the maximum degree of a node in the projection graph. If the maximum degree is low, the query is likely to be specialized. If there is no central node in the projection graph, the user will likely specialize the query. On the other hand, generalizations occur when the largest connected component of projection graph is large (more than 10 nodes, for the top 20 results) and where nodes are close together (low average path length in connector graph).

11.6 Connections to prior research

In prior research, investigators have explored the use of query terms to identify properties and relationships among specific parts of the web graph. In the HITS work by Kleinberg [Kleinberg, 1999a], eigenvectors are used to identify authoritative nodes using the notion of focused subgraphs defined by a query and associated links, and mutually reinforcing hubs and authorities. The work is similar to ours in that it extracts and operates on a query-defined subset of the web graph. In contrast to methods we have presented, HITS calculates a single property of a node (the corresponding component of the 1st singular vector of graph adjacency matrix), which is used to rank search results. We use a much wider range of graphical features that characterize whole subgraphs.

Variants of PageRank that work with subgraphs of the web have also been explored in prior research. This work includes explorations of domain-specific or person-specific PageRank [Richardson and Domingos, 2002a, Haveliwala, 2002], and on the use of non-random jump vectors for personalization [Jeh and Widom, 2003]. Related work on identifying web spam has examined methods for propagating from trusted-pages [Wu et al., 2006]. Recent work by Nie *et al.* [Nie et al., 2006] focused on eigenvectors or counts to set node priors.

In the content domain, Cronen-Townsend et al. [Cronen-Townsend et al., 2002] have looked at techniques for predicting the quality of results (what they call *query difficulty*) by computing the entropy between the language model for the results and the collection as a whole, but they do not consider any graphical properties. Several efforts have combined links and content in different ways. For example, Chakrabarti et al. [Chakrabarti et al., 1999] use link information on classes of neighbors to improve text classification accuracy of a target page. Dean and Henzinger [Dean and Henzinger, 1999] use links and content to find related pages, making use of information about the simple existence of links, rather than the rich topological characteristics of subgraphs that we represent and exploit. There has been research on considering multiple objectives, for example examining relationships among papers, authors, and institutions. In this work, relationships have been computed globally, based on one or more sets of similarity measures [Xi et al., 2004, Nie et al., 2005].

Related work also includes research on citation analysis, including Garfield's early work on the impact factor [Garfield, 1972], and later refinements by Pinski and Narin [Pinski and Narin, 1976]. Vassilvitskii and Brill have recently explored the use of distance (and direction) in the web graph for relevance feedback in web search [Vassilvitskii and Brill, 2006]. Minkov *et al.* [Minkov et al., 2006] have examined contextual search and name disambiguation in email messages using graphs, employing random walks on graphs to disambiguate names. In the context of machine learning on graphs, researchers have sought to predict the labels of vertices in a graph, given the known labels of vertices in training data [Kondor and Lafferty, 2002, Leskovec et al., 2005c, Brank and Leskovec, 2003]. A similar formulation has recently been explored in the context of kernel methods [Kashima et al., 2003], and approaches based on extracting subgraphs as features [Kudo et al., 2004].

In contrast to previous efforts, we examine a broad set of graph-theoretic properties of subgraphs, rather than, for example, only examining a single feature such as the eigenvalue associated with individual nodes. The richer characterization of the topological properties of subgraphs as a whole—or of individual nodes relative to the subgraph—allows us to investigate the discriminability of multiple features, to learn models for diverse classification goals, and, more generally, to provide useful analytical tools for exploring the web.

11.7 Summary and directions

We presented a methodology for learning predictive models and attributes from a rich set of topological characteristics of sets of web pages, based on their projection on the larger web graph. First, we considered patterns of connectivity among the set of pages returned as search results by projecting them onto the web graph, and analyzed the topological properties of the induced subgraphs. Using these graph-theoretic features, we performed machine learning to build classifiers that discriminate good and poor sets of results. Then, we learned models that predict user behavior when reformulating queries, including whether queries are likely to be reformulated, and the nature of the reformulation. The experimental results for the two problem domains highlight the potential value of employing contextual subgraphs for understanding search-related tasks.

The web-projection method is scalable as demonstrated by the sizes of datasets used in our analyses. Calculating the graph features is fast since projected graphs are fairly small. The most computationally expensive operation is obtaining the query connection graph. Here a shortest-path algorithm is performed to connect the components of query projection graph. If the components are far apart and the graph is densely linked, the shortest-path algorithm will have to traverse most of the graph. In rare cases, this problem arises in the domain graph and, the algorithm can take up a minute to complete. On average, however, less than three seconds were required to project a query and produce the projection and connection graphs. In case of URL graph, which is not as densely connected, we did not see visible delays in the production of the graph projections.

The method of projecting sets of results on the underlying graph of the web and then using machine learning with graph-theoretic features can be applied in many settings. For example, prior work has noted that spam web pages have distinct linkage patterns. We can use the web-projection method to identify either sets of results that are likely to contain many spam pages, or more specifically to identify pages that are likely to be spam (using graphical features of individual nodes as we will discuss next).

Applying our ideas to enhance ranking is especially interesting as it involves looking at features of individual nodes relative to a subset, which is interesting and has wider applicability than ranking itself. There are several ways one could approach this opportunity. One of the approaches we have been pursuing considers the inclusion of additional node-centric features from the projection and connection graphs, including those describing the position of the node with regard to the rest of the graph. These node-centric features can then be used to train existing ranking algorithms (*e.g.*, RankNet). Another application of our work is to discover missing human relevance scores. Given a projection graph where we have human relevance judgments for a few nodes, we would like to predict the judgments that would be assigned to the rest of the nodes.

Another promising direction for research is exploring the role of the connector nodes. We used these nodes to connect disconnected components of the projection graph. However, we observed that the connector nodes often become hubs holding the network together. As an example, see Figure 11.5(b) where the central connector node is a hub. However, there are also cases where no such patterns emerge, (*e.g.*, Figure 11.6(b)). To explore these questions, a better understanding of the quality of our heuristics to connect components of projection graph is needed. In the analyses described, we used a greedy heuristic that randomly chooses connections from the set of competing paths of the same length. We do not yet understand the sensitivity this heuristic in the overall procedure for selecting nodes.

With regard to modeling users and their queries, we are excited about exploring clusters of queries and query transitions based on graphical properties of their projection and connection graphs. The rich eviden-

tial patterns provided by graph-theoretic features promise to be valuable in describing different characteristics of queries, determining the classes of queries, and mapping query transitions to these classes.

One could also apply the ideas for modeling web searchers' behaviors in other ways. Web projections might be used to construct predictive models that infer paths that the searchers will likely take when reformulating a query from an initial query. Predictive models could be used to suggest likely query reformulations, specializations, generalizations and avoid transitions that would not likely lead to good sets of results, as captured by the graphical properties of projections associated with the sets.

Other applications include using the graph projections to explore the dynamics and evolution of the web, where models learned from features capturing topology and topological dynamics could help us to understand how sites and pages that created or removed over time relate to the rest of the web. Such models promise to be valuable in predicting the conceptual links and quality of new content and sites.

We believe that the represented work captures a starting point with the use of web projections. We found that the methods complement existing textual and graphical analyses and provoke tantalizing questions and interesting directions for future research in web search and retrieval, including efforts on enhancing result quality and on better understanding and supporting human behavior. We hope that the ideas will be of value to others pursuing insights about the nature and use of graphical properties of the web.

Part 3 – Large data: Conclusion

Working with large data presents several engineering, systems and implementation challenges. Moreover, it forces us to develop scalable algorithms and tools that scale to large data and allow for measurement and analysis. As these challenges are worth exploring for themselves there is a twofold benefit. In addition large data gives us opportunities to make observations and models of phenomena that is practically invisible on smaller scales.

Observations: First, we have reviewed a set of results stemming from the analysis of the communication patterns of all people using a popular IM system that provides a worldwide lens onto aggregate human behavior. We described capturing high-level communication activities and demographics in June 2006 with more than 30 billion conversations among 240 million people. Our communication network is *largest social network* analyzed to date. The planetary-scale network allowed us to explore dependencies among user demographics, communication characteristics, and network structure like the *6.6 degrees of separation*. Working with such a massive dataset allowed us to test hypotheses such as the degrees of separation among people across the entire world.

Models: Next, we investigated statistical properties of community structure in large real-world social and information networks. We find communities exist *only up to size scale* ≈ 100 nodes and then “blended in” an *intermingled network core*. This is interesting as it agrees well with the Dunbar’s number[Dunbar, 1998] of 150 which gives an upper bound on the human community size. As most of previous works on community detection focused on small networks they did not hit the Dunbar’s limit. By analyzing networks of millions of nodes, we discovered that community structure in these networks is very different than what we expected from the literature and from commonly-used models. Our work opens several new questions about the structure of large social and information networks in general, and it has implications for the use of graph partitioning algorithms on real-world networks and for detecting communities in them.

Algorithms: Last, we showed how in large networks local network structure can be used to make predictions about the whole network. We presented a methodology for learning *predictive models* and attributes from a rich set of topological characteristics of sets of web pages, based on their projection on the larger web graph. Using these context sensitive subgraphs we learned models that predict *search result quality*, *web search spam*, and *user behavior* when reformulating queries, including whether queries are likely to be reformulated, and the nature of the reformulation.

Part IV

Conclusion and future directions

Chapter 12

Conclusion

The ubiquity and the emergence of the web and rich social computing applications provide computer science with a unique opportunity to not only study but also design and build such complex computing systems and applications. Indeed, we are not just observers that measure and model, but we can also design, create and impose rules and incentives on such systems. Via systems like Facebook or Google that are used by millions of people we can influence each individual's behavior. However, decisions we make at the level of an individual user will have global effects on a system like the Facebook and the structure of their social network. So, it is important to understand how such systems work, what is their structure, and understand the global consequences of our micro-level decisions.

Our thesis presents a combination of (a) empirical work, measurements and experiments, (b) explanatory modeling and analysis of mathematical models, (c) design of algorithmic and machine learning tools. This naturally closes the loop between design and engineering on one hand, and empirical measurement and modeling on the other hand.

The research focus of this thesis is to analyze and model the structure, evolution and dynamics of large real-world networks. Our contributions so far are the following: we discovered novel structural properties of time evolving networks, namely the Densification Power Law and Shrinking Diameters. We developed simple models that explain the behavior we observed. We introduced the Kronecker graph model that can accurately mimic the structural properties of real networks, and developed algorithms for efficiently estimating its parameters.

On the information cascade side, we presented analyses of information propagation in large blog and product recommendation networks. We showed that cascade frequency does not simply decrease monotonically for denser cascade subgraphs, but rather reflects more subtle features of the domain in which the behavior is propagating. We also developed scalable algorithms for cascade and outbreak detection in networks that provably achieve near optimal solutions.

Last, we also showed how working with large datasets gives us opportunities to observe phenomena that are practically invisible at small scales. We demonstrated this by analyzing the planetary scale social network of Microsoft Messenger, and made novel observations about network community structure.

In the long run, outside the scope of this thesis, we would like to build tools for modeling the evolution of large networks both on a global scale and also on the micro-scale of nodes or small communities. We want to study how information flows through the network and how local communities or groups influence

Thesis part	Steps of the thesis		
	1: Observations	2: Models	3: Algorithms
Part 1: Network evolution	chapter 3	chapter 4	chapter 5
Part 2: Network cascades	chapter 6	chapter 7	chapter 8
Part 3: Large data	chapter 9	chapter 10	chapter 11

Table 12.1: Structure of the thesis with references to the chapters.

the global network and its evolution. Ideally, we want to bring these two views together, so that we can describe the evolution of the network as a whole, and at the same time also of its subparts.

Next, we give a summary of contributions and our vision for future research.

12.1 Summary of contributions

We summarize our contributions by grouping them by the columns of the thesis structure as summarized in table 12.1. The thesis adheres to the following three steps. (1) We start with describing novel empirical observations and studies. (2) These observations led us to revisit existing models and either improve on them or invent completely new models that incorporate our empirical findings. (3) Given intuitions coming from the models, we then develop novel algorithms that help us harness the empirical observations we made.

Observations:

- We discovered the network *densification* and *shrinking diameter* that influenced the thinking about fundamental structural properties of networks varying over time.
- We analyzed the properties of the planetary MSN Messenger social network, the *largest social network* examined to date and found the “*6.6 degrees of messaging*”, *i.e.*, that people are on average separated by only 6.6 friendships.
- Our work on the shape of the human adoption curve and cascades in viral marketing and the blogosphere was the first to observe, measure and analyze cascading behavior in a large real-world setting. We also found that the human adoption curve follows *diminishing returns* (Figure 6.8).

Models:

- We developed the Kronecker graph model, which is a *mathematically tractable* model of network generation and evolution. Moreover, the Kronecker graph is the first model that is able to capture *several* temporal (densification, shrinking diameter) and static (heavy tailed degree distributions, and other power laws) network properties at the same time.
- By observing the exact edge creation sequence of large social networks, we also developed a Triad Closing model that first *fully* specifies the network evolution: node arrival process, edge arrival process and the edge creation process.

Algorithms:

- We developed KRONFIT, the algorithm for estimating parameters of a Kronecker graphs model. Naive parameter estimation takes $O(N!N^2)$ time, while our approach scales *linearly* $O(E)$, which allows us to fit large graphs with millions of nodes and edges. Once the parameters are estimated they can be naturally used to generate synthetic realistic-looking networks.
- We also developed the CELF algorithm for sensor placement to detect disease and information outbreaks in networks. We proved that CELF placements are near optimal, and obtained data dependent bounds that show our obtained solutions are at $\approx 90\%$ of NP-hard to compute optima, while being *700 times faster* than a simple non-optimal greedy algorithm.

12.2 Vision for the future

Our long-term research goal is to harness large-scale networks to understand, predict, and ultimately, enhance social and technological systems. We would like to create explanatory and predictive models of actions of large groups of people and societies, and biological and technological systems. Although the actions of a particular individual or component may be too difficult to model, machine learning and statistics can be applied to large groups or ensembles, which can yield effective models with the ability to predict the flow of future events. Based on our recent results and research experience, we believe that the study of large networks is a promising approach to developing such understandings, as graphs capture local dependencies, and also reveal large-scale structure and phenomena arising from the multitude of local interactions. Seemingly “random” local behavior can propagate to the macro scale where global regularities and patterns emerge, *e.g.*, power law degree distributions and small-diameters.

On the way to achieving this long-term goal, our research consists of:

- (1) Analyzing theoretical models of network structure and evolution.
- (2) Developing statistical machine learning models and algorithms to efficiently estimate the network properties and parameters from data.
- (3) Working with massive datasets of gigabyte and terabyte scale, as certain behaviors and patterns are observable only when the amount of data is large enough.
- (4) Working with richer types of networks of different modalities and with different types of data attached to nodes and edges to capture the complex behaviors in finer detail that will allow for novel observations and models.

The long-term goal of our research is to build and harness models of natural and synthetic systems to make predictions about future events. We believe that it will be feasible in the long-term to predict events and overall dynamics of the behavior of networked systems, such as large groups of people, web, communication, and biological networks. The idea is that actions of an individual are too hard to model but machine learning can be applied to large groups to predict the general flow of future events. We believe the right approach is through networks where local dependencies and behavior propagate to global patterns and trends. The key is to connect local to global, complement the topology information with other types of data, and choose the right scale where micro behaviors propagate to macroscopic structure.

In our thesis research, we made several steps towards this long-term goal. We now better understand microscopic and macroscopic network evolution and models that connect the two. Moreover, we can

efficiently fit the network models to the data and predict prior and future states of the network. We also have a better understanding of how information and influence propagate over the network, what the traces of propagation are, and how to select influential nodes or detect disease outbreaks.

On the road towards the long-term goal, our research will focus across three dimensions: (1) structure and dynamics of networks, designing networked systems and influencing their evolution, (2) encompassing richer types of networked data, and (3) scaling up the analyses to massive datasets and internet-scale computing.

12.2.1 Medium term goals

We first present medium term goals for future work that builds on the work presented in the thesis.

Network structure and communities

The online world is a rich testbed for our research as web media and social networking sites are used by hundreds of millions of users and contain very detailed traces of human social activity, people's profiles, interests, groups, etc. We want to understand how network topology, user profiles, and past actions determine the future of particular groups or events, and, more abstractly, how links arise and decay. We want to define and explore the "*health*" of a social network or a community, which is important, for e.g., social network web sites like Facebook, where it is crucial that the social network be healthy and "organic". Intuitively, in healthy networks users get utility from links and communities don't diminish, separate or die. A natural next step is then to suggest actions and mechanisms that one could apply to improve the network health. In this context, we are collaborating with LinkedIn and Facebook; we plan to perform large-scale data mining and machine learning for social network analysis.

Actively analyzing and influencing the network

Beyond simply observing and characterizing a network, one can try to influence the activities and overall evolution of a network. For example, we can explore mechanisms that would help a community to evolve in a healthy manner and continue to grow organically. The key here is to make a step from passive observations and modeling to actively trying to steer and influence the development of the network or community. Basically, one would hope to run live real-time experiments over large populations to test hypotheses, build models and test their predictive powers. As the experiments run they would actively change the behavior of the users, which in turn would change the network structure. Explorations in this realm could lead to strategies that help networked communities to survive and serve as richer resources for people. For example, one such way is to study *how to introduce incentive mechanisms in networks?*. Incentives are necessarily local in that they stimulate behavior of each individual user, but the goal here is to achieve a global change in the network structure. For this reason it is important to study how local micro behaviors propagate to global macro scale.

Information propagation

Another important aspect is the online media and information propagation: *How do people (sites) consume and alter information, and how do they influence propagation?* For example, information from the New York Times probably spreads in a different way than Slashdot posts. Moreover, when a story is posted on Slashdot, it is given a special boost. A certain community starts discussing it, which further diffuses the story. Most of our knowledge about such phenomena is either based on anecdotal evidence or small studies. On the other hand we wish to simultaneously analyze millions of news sources and build

predictive models of such behavior. Natural case studies here are predicting opinion formation and its outcomes, finding trendsetters, predicting election results, the success of a new product, and the evolution of content on Wikipedia.

12.2.2 Long-term goals

Last, we present the long-term goals of our research.

Massive data and scalability

One of our primary research agendas focuses on large scale data and computing architectures for *massive data* manipulation and analysis. We plan to explore *map-reduce* type programming abstractions for large scale computing and extend our software library to distributed “share nothing” architectures. The question here is what kinds of analyses are suitable for such architectures, and how to parallelize data mining and machine learning algorithms to scale to thousands of machines. Here, we are collaborating with the CMU Parallel Data Lab and Yahoo Research, who recently gave us access to a 4,000 processor Hadoop cluster. This line of research will allow us to perform near real-time analysis of planetary and internet scale data and find patterns that are practically unobservable at smaller scales.

Richer types of graphs

Most algorithms and models today work on simple undirected graphs. We plan to extend generative models and mining algorithms to graphs with multiple edges between pairs of nodes and to graphs with different types of nodes and weights or attributes on nodes and edges. Incorporating other *data modalities* like textual information, and historical and communication data, will allow for richer and more accurate models.

These steps capture our sense for the beginning of an evolving research framework that will allow us to tackle these challenging problems in a unique way. Our research on networks is theoretically grounded and spans several areas of computer science as diverse as machine learning, theory and systems. Implications of our research have direct applications well beyond computer science — to social sciences, physics, economics and marketing. In short, the vision for future work is on patterns, scalability and predictive modeling in large real networked systems.

Appendix A

Appendix

A.1 Table of symbols

Table A.1 defines the symbols used in the thesis. For reader's convenience each of the chapters of the thesis also lists symbols related to that particular chapter.

SYMBOL	DESCRIPTION
\mathcal{A}	Set of elements, $\mathcal{A} = \{a_1, \dots, a_n\}$
\mathcal{E}	Edge set of a graph
\mathcal{G}	Graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
\mathcal{I}	Set of all possible outbreaks (set of all possible cascades)
$\mathcal{P} = \mathcal{P}_1^{[k]} = \mathcal{P}_k$	k^{th} Kronecker power of Stochastic Kronecker initiator matrix \mathcal{P}_1
\mathcal{V}	Vertex set of a graph
$C_{a,b}$	Set of all conversations between users of age a and b
$C_{g,h}$	Set of all conversations between users of genders g and h
D	Diameter of a graph as defined in def. 2.1.1
D^*	Effective diameter of the graph as defined in def. 2.1.3
E	Number of edges in a graph
E_1	Number of edges in Kronecker initiator K_1
$E(t)$	Number of edges in a graph at time t
E_c	Number of edges in largest component
E_h	Number of edges that at the time of creation span h hop path
G	Graph or graph adjacency matrix
G_t	Graph composed of nodes and edges that arrived before time t
$G \otimes H$	Kronecker product of adjacency matrices of graphs G and H
H_Γ	Height of the tree Γ
K	Kronecker graph (synthetic estimate of graph G)
K_1	Initiator of a Kronecker Graph
$K_1^{[k]} = K_k = K$	k^{th} Kronecker power of K_1

Continued on next page...

SYMBOL	DESCRIPTION
$K_1[i, j]$	Entry at row i and column j of K_1
N	Number of nodes in a graph
N_1	Number of nodes in Kronecker initiator K_1
$N(t)$	Number of nodes in a graph at time t
N_c	Number of nodes in the largest weakly connected component of a graph
N_s	Number of senders of recommendations
N_r	Number of recommendation receivers
N_t	Size of the cascade at time t
$R(\mathcal{P})$	Realization of a Stochastic Kronecker graph \mathcal{P}
$R(\mathcal{A})$	Reward for detecting a cascade, <i>i.e.</i> , penalty reduction $R(\mathcal{A}) = \pi(\emptyset) - \pi(\mathcal{A})$
S	Set of nodes (smaller side of the cut)
T	Time span of a graph
$T(i, s)$	Time it takes an outbreak (cascade) i to reach node s
a	Densification power law exponent, $E(t) \propto N(t)^a$
$a_t(u)$	Age of a node u at time t , $a_t(u) = t - t(u)$
b	Community hierarchy branching factor in Community Guided Attachment
b_r	Purchases per recommender, $b_r = (b_b + b_e)/r$
b_b	Number of purchases with buy-bit turned on
b_e	Number of purchases as determined via buy-edges
c	Difficulty Constant in Community Guided Attachment
c_{av}	Average number of people recommending a product
$c(s)$	Cost of monitoring (placing a sensor, reading a blog) node s
$c(\mathcal{A})$	Cost of placement \mathcal{A} , $c(\mathcal{A}) = \sum_{s \in \mathcal{A}} c(s)$
d	Node degree, <i>i.e.</i> , number of adjacent nodes
d_i	Duration of i^{th} conversation on instant messenger
$d_t(u)$	Degree of node u at time t
$d(u)$	Final degree of node u (number of edges incident to node u)
\bar{d}	Average node degree in a graph
d_{max}	Maximum node degree in a graph
$e = (u, v)$	Edge connecting nodes u and v in a graph
e_t	t^{th} edge in a graph
$f(h)$	Difficulty Function in Community Guided Attachment
h	Number of hops, path length, distance
$h(u, v)$	Length of the shortest path between nodes u and v
$h_\Gamma(v, w)$	Least common ancestor height of leaves v, w in Γ
$l(\Theta)$	Log-likelihood of parameters Θ generating real graph G , $\log P(G \Theta)$
l_i	Geographical distance between the a pair of users in conversation i
m_i	Number of exchanged messages in i^{th} conversation
$m_{u,i}$	Number of exchanged messages in i^{th} conversation of user u
n_p	Number of products
p	Forest Fire forward burning probability
p_b	Forest Fire backward burning probability
p_{ij}	Probability of edge (i, j)

Continued on next page...

SYMBOL	DESCRIPTION
$p_{ij} = \mathcal{P}_k[i, j]$	Probability of an edge (i, j) in \mathcal{P}_k , entry at row i and column j of \mathcal{P}_k
$p_e(d)$	Probability of new edge linking to node of degree d
$p_l(a)$	Node lifetime distribution, <i>i.e.</i> , prob. of node being alive at age a
p_m	Median product price
p_t	Probability of a recommendation causing a purchase
r_{p1}	Average number of reviews per product in 2001–2003
r	Forest Fire ratio of backward and forward probability, $r = p/p_b$
r_r	Number of recommendations
r_c	Number of recommendation in the largest component
s_k	Location with highest marginal reward or benefit/cost ratio
$t(e)$	Time of creation of an edge e
$t(v)$	Time when node v joined the network (created its first edge)
t_u	Time when post u was published
$t_i(u)$	Time of creation of i^{th} edge of a node u
t_{ij}	Time of j^{th} login of a user
t_{oj}	Time of j^{th} logout of a user
$ts_{u,i}$	Start time of i^{th} conversation of user u
$te_{u,i}$	End time of i^{th} conversation of user u
v_{av}	Average star rating of a product or set of products
u, v, w	Nodes in a graph
w_k	Weight (expected degree) of node k in a graph
Γ	Community hierarchy (tree) in Community Guided Attachment
Δ	Propagation delay on edge (u, v) , $\Delta = t_u - t_v$
$\Theta = \mathcal{P}_1$	Stochastic Kronecker initiator
$\hat{\Theta}$	Parameter values at maximum likelihood
$\Phi(k)$	Minimum conductance of all sets on k elements
γ	Power law degree exponent, $p(d) \propto d^{-\gamma}$
$\delta_u(d)$	Edge gap, time between d^{th} and $d + 1^{th}$ edge of u , $\delta_u(d) = t_{d+1}(u) - t_d(u)$
δ_s	Marginal reward (gain), $\delta_s = R(\mathcal{A} \cup s) - R(\mathcal{A})$
$\theta_{ij} = \mathcal{P}_1[i, j]$	Entry at row i and column j of \mathcal{P}_1
κ	Decay exponent in $E_h \propto \exp(-\kappa h)$
λ	Node lifetime distribution parameter (exponential distribution)
$\pi(\mathcal{A})$	Expected penalty over all possible outbreaks (cascades) \mathcal{I}
σ	Permutation defining node correspondences, <i>i.e.</i> , maps node ids of G to those of \mathcal{P}
$\phi(S)$	Conductance of set S as defined in def. 10.2.1
ω	Proportion of times SwapNodes permutation proposal distribution is used

Table A.1: Table of all symbols used in this thesis. Symbols are sorted alphabetically.

A.2 Datasets and basic statistics

Last, tables A.2, A.3 and A.4 list and briefly describe the network datasets used in this thesis. We also report some of the basic properties and statistics of each of the datasets.

Network	N	E	N_b	E_b	\bar{d}	\tilde{d}	\bar{C}	D	\bar{D}	Description
Social networks										
DELICIOUS	147,567	301,921	0.40	0.65	4.09	48.44	0.30	24	6.28	del.icio.us collaborative tagging social network
EPINIONS	75,877	405,739	0.48	0.90	10.69	183.88	0.26	15	4.27	Who-trusts-whom graph of opinions.com [Richardson et al., 2003]
FLICKR	404,733	2,110,078	0.33	0.86	10.43	442.75	0.40	18	5.42	Flickr photo sharing social network [Kumar et al., 2006]
LINKEDIN	6,946,668	30,507,070	0.47	0.88	8.78	351.66	0.23	23	5.43	Social network of professional contacts [Leskovec et al., 2008a]
LIVEJOURNAL01	3,766,521	30,629,297	0.78	0.97	16.26	111.24	0.36	23	5.55	Blogging community friendship network [Backstrom et al., 2006]
LIVEJOURNAL11	4,145,160	34,469,135	0.77	0.97	16.63	122.44	0.36	23	5.61	Blogging community friendship network [Backstrom et al., 2006]
LIVEJOURNAL12	4,843,953	42,845,684	0.76	0.97	17.69	170.66	0.35	20	5.53	Blogging community friendship network [Backstrom et al., 2006]
MESSENGER	1,878,736	4,079,161	0.53	0.78	4.34	15.40	0.09	26	7.42	Instant messenger social network [Leskovec et al., 2008b]
EMAIL-ALL	234,352	383,111	0.18	0.50	3.27	576.87	0.50	14	4.07	Research organization email network [Leskovec et al., 2007b]
EMAIL-INOUT	37,803	114,199	0.47	0.82	6.04	165.73	0.58	8	3.74	(email has to be sent both ways) [Leskovec et al., 2007b]
EMAIL-INSIDE	986	16,064	0.90	0.99	32.58	74.66	0.45	7	2.60	(only addresses inside the organization) [Leskovec et al., 2007b]
EMAIL-ENRON	33,696	180,811	0.61	0.90	10.73	142.36	0.71	13	3.99	Enron email dataset [Klimt and Yang, 2004]
ANSWERS	488,484	1,240,189	0.45	0.78	5.08	251.78	0.11	22	5.72	Yahoo Answers social network [Leskovec et al., 2008a]
ANSWERS-1	26,971	91,812	0.56	0.87	6.81	59.17	0.08	16	4.49	Cluster 1 from Yahoo Answers
ANSWERS-2	25,431	65,551	0.48	0.80	5.16	56.57	0.10	15	4.76	Cluster 2 from Yahoo Answers
ANSWERS-3	45,122	165,648	0.53	0.87	7.34	417.83	0.21	15	3.94	Cluster 3 from Yahoo Answers
ANSWERS-4	93,971	266,199	0.49	0.82	5.67	94.48	0.08	16	4.91	Cluster 4 from Yahoo Answers
ANSWERS-5	5,313	11,528	0.41	0.73	4.34	29.55	0.12	14	4.75	Cluster 5 from Yahoo Answers
ANSWERS-6	290,351	613,237	0.40	0.71	4.22	57.16	0.09	22	5.92	Cluster 6 from Yahoo Answers
Information (citation) networks										
CIT-PATENTS	3,764,105	16,511,682	0.82	0.96	8.77	21.34	0.09	26	8.15	Citation network of all US patents [Leskovec et al., 2005b]
CIT-HEP-PH	34,401	420,784	0.96	1.00	24.46	63.50	0.30	14	4.33	Citations between arXiv hep-th papers [Gehrke et al., 2003]
CIT-HEP-TH	27,400	352,021	0.94	0.99	25.69	106.40	0.33	15	4.20	Citations between arXiv hep-ph papers [Gehrke et al., 2003]
BLOG-NAT05-6M	29,150	182,212	0.74	0.96	12.50	342.51	0.24	10	3.40	Blog citation network (6 months of data) [Leskovec et al., 2007d]
BLOG-NAT06ALL	32,384	315,713	0.87	0.99	19.50	153.08	0.20	18	3.94	Blog citation network (1 year of data) [Leskovec et al., 2007d]
POST-NAT05-6M	238,305	297,338	0.21	0.34	2.50	39.51	0.13	45	10.34	Blog post citation network (6 months) [Leskovec et al., 2007d]
POST-NAT06ALL	437,305	565,072	0.22	0.38	2.58	35.54	0.11	54	10.48	Blog post citation network (1 year) [Leskovec et al., 2007d]
Collaboration networks										
ATA-IMDB	883,963	27,473,042	0.87	0.99	62.16	517.40	0.79	15	3.48	IMDB actor collaboration network from Dec 2007
CA-ASTRO-PH	17,903	196,972	0.89	0.98	22.00	65.70	0.67	14	4.21	Co-authorship in astro-ph of arxiv.org [Leskovec et al., 2005b]
CA-COND-MAT	21,363	91,286	0.81	0.93	8.55	22.47	0.70	15	5.36	Co-authorship in cond-mat category [Leskovec et al., 2005b]
CA-GR-QC	4,158	13,422	0.64	0.78	6.46	17.98	0.66	17	6.10	Co-authorship in gr-qc category [Leskovec et al., 2005b]
CA-HEP-PH	11,204	117,619	0.81	0.97	21.00	130.88	0.69	13	4.71	Co-authorship in hep-ph category [Leskovec et al., 2005b]
CA-HEP-TH	8,638	24,806	0.68	0.85	5.74	12.99	0.58	18	5.96	Co-authorship in hep-th category [Leskovec et al., 2005b]
CA-DBLP	317,080	1,049,866	0.67	0.84	6.62	21.75	0.73	23	6.75	DBLP co-authorship network [Backstrom et al., 2006]

Table A.2: Network datasets we analyzed. Statistics of networks we consider: number of nodes N ; number of edges E ; fraction nodes not in whiskers (size of largest biconnected component) N_b/N ; fraction of edges in biconnected component E_b/E ; average degree $\bar{d} = 2E/N$; second order average degree \tilde{d} ; average clustering coefficient \bar{C} ; diameter D ; and average path length \bar{D} .

Network	N	E	N_b	E_b	\bar{d}	\tilde{d}	\bar{C}	D	\bar{D}	Description
Web graphs										
Internet networks										
WEB-BERKSTAN	319,717	1,542,940	0.57	0.88	9.65	1,067.55	0.32	35	5.66	Stanford and Berkeley [Khalil and Liu, 2004]
WEB-GOOGLE	855,802	4,291,352	0.75	0.92	10.03	170.35	0.62	24	6.27	Web graph Google released in 2002 [Google, 2002]
WEB-NOTREDAME	325,729	1,090,108	0.41	0.76	6.69	280.68	0.47	46	7.22	Web graph of Uni. of Notre Dame [Albert et al., 1999]
WEB-TREC	1,458,316	6,225,033	0.59	0.78	8.54	682.89	0.68	112	8.58	Web graph of TREC WT10G web corpus [NIST, 2000]
Bi-partite networks										
IPTRAFFIC	2,250,498	21,643,497	1.00	1.00	19.23	94,889.05	0.00	5	2.53	IP traffic graph a single router for 24 hours
ATP-ASTRO-PH	54,498	131,123	0.70	0.87	4.81	16.67	0.00	28	7.78	Affiliation network of astro-ph [Leskovec et al., 2007d]
ATP-COND-MAT	57,552	104,179	0.65	0.79	3.62	10.54	0.00	31	9.96	Affiliation network of cond-mat [Leskovec et al., 2007d]
ATP-GR-QC	14,832	22,266	0.47	0.60	3.00	9.72	0.00	35	11.08	Affiliation network of gr-qc [Leskovec et al., 2007d]
ATP-HEP-PH	47,832	86,434	0.60	0.76	3.61	16.80	0.00	27	8.55	Affiliation network of hep-ph [Leskovec et al., 2007d]
ATP-HEP-TH	39,986	64,154	0.53	0.68	3.21	13.07	0.00	36	10.74	Affiliation network of hep-th [Leskovec et al., 2007d]
ATP-DBLP	615,678	944,456	0.49	0.64	3.07	13.61	0.00	48	12.69	DBLP authors-to-papers bipartite network
SPENDING	1,831,540	2,918,920	0.34	0.58	3.19	1,536.35	0.00	26	5.62	Users-to-keywords they bid [Leskovec et al., 2008c]
HW7	653,260	2,278,448	0.99	0.99	6.98	346.85	0.00	24	6.26	Downsampled advertiser-query bid graph
NETFLIX	497,959	100,480,507	1.00	1.00	403.57	28,432.89	0.00	5	2.31	Users-to-movies they rated. Netflix prize [Netflix, 2006]
QUERYTERMS	13,805,808	17,498,668	0.28	0.41	2.53	14.92	0.00	86	19.81	Users-to-queries they submit to a search engine
CLICKSTREAM	199,308	951,649	0.39	0.87	9.55	430.74	0.00	7	3.83	Users-to-visited URLs [Montgomery and Faloutsos, 2001]
Biological networks										
BIO-PROTEINS	4,626	14,801	0.72	0.91	6.40	24.25	0.12	12	4.24	Yeast protein interaction network [Colizza et al., 2005]
BIO-YEAST	1,458	1,948	0.37	0.51	2.67	7.13	0.14	19	6.89	Yeast protein interaction network data [Jeong et al., 2001]
BIO-YEASTP0.001	353	1,517	0.73	0.93	8.59	20.18	0.57	11	4.33	Yeast protein-protein interaction map [Qi et al., 2005]
BIO-YEASTP0.01	1,266	8,511	0.79	0.97	13.45	47.73	0.44	12	3.87	Yeast protein-protein interaction map [Qi et al., 2005]

Table A.3: Network datasets we analyzed. Statistics of networks we consider: number of nodes N ; number of edges E ; fraction nodes not in whiskers (size of largest biconnected component) N_b/N ; fraction of edges in biconnected component E_b/E ; average degree $\bar{d} = 2E/N$; second order average degree \tilde{d} ; average clustering coefficient \bar{C} ; diameter D ; and average path length \bar{D} .

Network	N	E	N_b	E_b	\bar{d}	\tilde{d}	\bar{C}	D	\bar{D}	Description
Nearly low-dimensional networks										
ROAD-CA	1,957,027	2,760,388	0.80	0.85	2.82	3.17	0.06	865	310.97	California road network
ROAD-USA	126,146	161,950	0.97	0.98	2.57	2.81	0.03	617	218.55	USA road network (only main roads)
ROAD-PA	1,087,562	1,541,514	0.79	0.85	2.83	3.20	0.06	794	306.89	Pennsylvania road network
ROAD-TX	1,351,137	1,879,201	0.78	0.84	2.78	3.15	0.06	1,064	418.73	Texas road network
POWERGRID	4,941	6,594	0.62	0.69	2.67	3.87	0.11	46	19.07	Western States Power Grid [Watts and Strogatz, 1998]
MANI-FACES7K	696	6,979	0.98	0.99	20.05	37.99	0.56	16	5.52	Faces (64x64 grayscale images) (connect 7k closest pairs)
MANI-FACES4K	663	3,465	0.90	0.97	10.45	20.20	0.56	29	8.96	Faces (connect 4k closest pairs) [Tenenbaum et al., 2000]
MANI-FACES2K	551	1,981	0.84	0.94	7.19	12.77	0.54	32	11.07	Faces (connect 2k closest pairs)
MANI-FACES10	698	6,935	1.00	1.00	19.87	25.32	0.51	6	3.25	Faces (connect every to 10 nearest neighbors)
MANI-FACES3K	698	2,091	1.00	1.00	5.99	7.98	0.45	9	4.89	Faces (connect every to 5 nearest neighbors)
MANI-FACES5	698	3,480	1.00	1.00	9.97	12.91	0.48	7	4.03	Faces (connect every to 3 nearest neighbors)
MANI-SWISS200K	20,000	200,000	1.00	1.00	20.00	21.08	0.59	103	37.21	Swiss-roll (connect 200k nearest pairs of nodes)
MANI-SWISS100K	19,990	99,979	1.00	1.00	10.00	11.02	0.59	162	58.32	Swiss-roll (connect 100k pairs) [Tenenbaum et al., 2000]
MANI-SWISS60K	19,042	57,747	0.93	0.96	6.07	7.03	0.59	243	89.15	Swiss-roll (connect 60k nearest pairs of nodes)
MANI-SWISS10	20,000	199,955	1.00	1.00	20.00	25.38	0.56	10	5.47	Swiss-roll (every node connects to 10 nearest neighbors)
MANI-SWISS5	20,000	99,990	1.00	1.00	10.00	12.89	0.54	13	8.34	Swiss-roll (every node connects to 5 nearest neighbors)
MANI-SWISS3	20,000	59,997	1.00	1.00	6.00	7.88	0.50	17	6.89	Swiss-roll (every node connects to 3 nearest neighbors)
IMDB Actor-to-Movie graphs										
ATM-IMDB	2,076,978	5,847,693	0.49	0.82	5.63	65.41	0.00	32	6.82	Actors-to-movies graph from IMDB (imdb.com)
IMDB-TOP30	198,430	566,756	0.99	1.00	5.71	18.19	0.00	26	8.32	Actors-to-movies graph heavily preprocessed
IMDB-RAW07	601,481	1,320,616	0.54	0.79	4.39	20.94	0.00	32	8.55	Country clusters were extracted from this graph
IMDB-FRANCE	35,827	74,201	0.51	0.76	4.14	14.62	0.00	20	6.57	Cluster of French movies
IMDB-GERMANY	21,258	42,197	0.56	0.78	3.97	13.69	0.00	34	7.47	German movies (to actors that played in them)
IMDB-INDIA	12,999	25,836	0.57	0.78	3.98	31.55	0.00	19	6.00	Indian movies
IMDB-ITALY	19,189	37,534	0.55	0.77	3.91	11.66	0.00	30	6.91	Italian movies
IMDB-JAPAN	15,042	34,131	0.60	0.82	4.54	16.98	0.00	19	6.81	Japanese movies
IMDB-MEXICO	13,783	36,986	0.64	0.86	5.37	24.15	0.00	19	5.43	Mexican movies
IMDB-SPAIN	15,494	31,313	0.51	0.76	4.04	14.22	0.00	28	6.44	Spanish movies
IMDB-UK	42,133	82,915	0.52	0.76	3.94	15.14	0.00	23	7.04	UK movies
IMDB-USA	241,360	530,494	0.51	0.78	4.40	25.25	0.00	30	7.63	USA movies
IMDB-WGERMANY	12,120	24,117	0.56	0.78	3.98	11.73	0.00	22	6.26	West German movies
Amazon product co-purchasing networks										
AMAZON0302	262,111	899,792	0.95	0.97	6.87	11.14	0.43	38	8.85	Amazon products from 2003 03 02 [Clauset et al., 2004]
AMAZON0312	400,727	2,349,869	0.94	0.99	11.73	30.33	0.42	20	6.46	Amazon products from 2003 03 12 [Clauset et al., 2004]
AMAZON0505	410,236	2,439,437	0.94	0.99	11.89	30.93	0.43	22	6.48	Amazon products from 2003 05 05 [Clauset et al., 2004]
AMAZON0601	403,364	2,443,311	0.96	0.99	12.11	30.55	0.43	25	6.42	Amazon products from 2003 06 01 [Clauset et al., 2004]
AMAZONALL	473,315	3,505,519	0.94	0.99	14.81	52.70	0.41	19	5.66	Amazon products (all 4 graphs merged) [Clauset et al., 2004]
AMAZONALLPROD	524,371	1,491,793	0.80	0.91	5.69	11.75	0.35	42	11.18	Products (all products, source+target) [Leskovec et al., 2007a]
AMAZONSRCPROD	334,863	925,872	0.84	0.91	5.53	11.53	0.43	47	12.11	Products (only source products) [Leskovec et al., 2007a]

Table A.4: Network datasets we analyzed. Statistics of networks we consider: number of nodes N ; number of edges E ; fraction nodes not in whiskers (size of largest biconnected component) N_b/N ; fraction of edges in biconnected component E_b/E ; average degree $\bar{d} = 2E/N$; second order average degree \tilde{d} ; average clustering coefficient \bar{C} ; diameter D ; and average path length \bar{D} .

Bibliography

- J. Abello. Hierarchical graph maps. *Computers & Graphics*, 28(3):345–359, 2004.
- J. Abello, A. L. Buchsbaum, and J. Westbrook. A functional approach to external graph algorithms. In *Proceedings of the 6th Annual European Symposium on Algorithms*, pages 332–343, 1998.
- J. Abello, Panos M. Pardalos, and M. G. C. Resende. *Handbook of massive data sets*. Kluwer Academic Publishers, 2002.
- L. A. Adamic. Zipf, power-law, pareto – a ranking tutorial. <http://www.hpl.hp.com/research/idl/papers/ranking>, 2000.
- L. A. Adamic and N. Glance. The political blogosphere and the 2004 u.s. election: divided they blog. In *LinkKDD '05: Proceedings of the 3rd international workshop on Link discovery*, pages 36–43, 2005.
- E. Adar and L. A. Adamic. Tracking information epidemics in blogsphere. In *Web Intelligence*, pages 207–214, 2005.
- E. Adar, L. Zhang, L. A. Adamic, and R. M. Lukose. Implicit structure and the dynamics of blogsphere. In *Workshop on the Weblogging Ecosystem*, 2004.
- S. E. Ahnert and T. M. A. Fink. Clustering signatures classify directed networks. *Physical Review E*, 78(3):036112, 2008.
- W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *STOC '00: Proceedings of the 32nd annual ACM symposium on Theory of computing*, pages 171–180, 2000.
- W. Aiello, F. R. K. Chung, and L. Lu. A random graph model for power law graphs. *Experimental Mathematics*, 10:53–66, 2001.
- E. M. Airoldi and K. M. Carley. Sampling algorithms for pure network topologies: a study on the stability and the separability of metric embeddings. *SIGKDD Explorations*, 7(2):13–22, 2005.
- E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2007.
- R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.
- R. Albert, H. Jeong, and A.-L. Barabási. Diameter of the world-wide web. *Nature*, 401:130–131, September 1999.
- R. Albert, H. Jeong, and A.-L. Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378, 2000.
- C. Allen. Life with alacrity: The Dunbar number as a limit to group sizes, 2004.
- U. Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461,

2007.

- J. I. Alvarez-Hamelin, L. Dall'Asta, A. Barrat, and A. Vespignani. Analysis and visualization of large scale networks using the k-core decomposition. In *ECCS '05: European Conference on Complex Systems*, 2005.
- R. Andersen and K. J. Lang. Communities from seed sets. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 223–232, 2006.
- R. Andersen, F. Chung, and K. J. Lang. Local graph partitioning using pagerank vectors. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 475–486, 2006.
- C. Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More*. 2006.
- R. M. Anderson and R. M. May. *Infectious diseases of humans: Dynamics and control*. 2002.
- Anonymous. Profiting from obscurity: What the “long tail” means for the economics of e-commerce. *Economist*, 2005.
- S. Arora and S. Kale. A combinatorial, primal-dual approach to semidefinite programs. In *STOC '07: Proceedings of the 39th annual ACM Symposium on Theory of Computing*, pages 227–236, 2007.
- S. Arora, E. Hazan, and S. Kale. $O(\sqrt{\log n})$ approximation to sparsest cut in $\tilde{O}(n^2)$ time. In *FOCS '04: Proceedings of the 45th Annual Symposium on Foundations of Computer Science*, pages 238–247, 2004a.
- S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC '04: Proceedings of the 36th annual ACM Symposium on Theory of Computing*, pages 222–231, 2004b.
- S. Asmussen. *Applied Probability and Queues*. Springer, 2003.
- D. Avrahami and S. E. Hudson. Communication characteristics of instant messaging: effects and predictions of interpersonal relationships. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 505–514, 2006.
- M. Babenko, J. Derryberry, A. Goldberg, R. Tarjan, and Y. Zhou. Experimental evaluation of parametric max-flow algorithms. In *WEA '07: Proceedings of the 6th Workshop on Experimental Algorithms*, pages 256–269, 2007.
- K. W. Back. Influence through social communication. *Journal of Abnormal and Social Psychology*, 46: 9–23, 1951.
- L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54, 2006.
- J. P. Bagrow and E. M. Boltt. Local method for detecting communities. *Physical Review E*, 72:046108, 2005.
- N. T. J. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. 2nd edition, 1975.
- P. Bak. *How Nature Works: The Science of Self-Organized Criticality*. Springer, September 1996.
- A.-L. Barabási. The origin of bursts and heavy tails in human dynamics. *Nature*, 435:207, 2005.
- A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
- A.-L. Barabási, E. Ravasz, and T. Vicsek. Deterministic scale-free networks. *Physica A*, 299:559–564, 2001.

- F. Bass. A new product growth for model consumer durables. *Management Science*, 15(5):215–227, 1969.
- V. Batagelj and M. Zaveršnik. Generalized cores. *ArXiv*, cs.DS/0202039, Feb 2002.
- I. Benjamini, G. Kozma, and N. Wormald. The mixing time of the giant component of a random graph. *ArXiv*, math/0610459, October 2006.
- J. Berry, W. E. Hart, C. E. Phillips, J. G. Uber, and J. Watson. Sensor placement in municipal water networks with temporal integer programming models. *J. Water Resources Planning and Management*, 132(4):218–224, 2006.
- I. Bezákova, A. Kalai, and R. Santhanam. Graph model selection using maximum likelihood. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 105–112, 2006.
- Z. Bi, C. Faloutsos, and F. Korn. The DGX distribution for mining massive, skewed data. In *KDD '01: Proceedings of the 6th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 17–26, 2001.
- G. Bianconi and A. L. Barabási. Competition and multiscaling in evolving networks. *Europhysics Letters*, 54:436–442, 2001.
- S. Bikhchandani, D. Hirshleifer, and I. Welch. A theory of fads, fashion, custom, and cultural change in informational cascades. *Journal of Political Economy*, 100(5):992–1026, October 1992.
- A. Blum, H. Chan, and M. Rwebangira. A random-surfer web-graph model. In *ANALCO '06: Proceedings of the 3rd Workshop on Analytic Algorithmics and Combinatorics*, 2006.
- S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4-5):175–308, 2006.
- B. Bollobas. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European J. of Combinatorics.*, 1(4):311–316, 1980.
- B. Bollobás. *Random Graphs*. Academic Press, London, 1985.
- B. Bollobas and O. Riordan. Mathematical results on scale-free random graphs. In S. Bornholdt and H. Schuster, editors, *Handbook of Graphs and Networks*, pages 1–37. 2003.
- B. Bollobas and O. Riordan. The diameter of a scale-free random graph. *Combinatorica*, 24(1):5–34, 2004.
- S. P. Borgatti and M. G. Everett. Models of core/periphery structures. *Social Networks*, 21(4):375–395, 2000.
- D. Bowman and D. Narayandas. Managing customerinitiated contacts with manufacturers: The impact on share of category requirements and word-of-mouth behavior. *Journal of Marketing Research*, 38(3): 281 – 297, August 2001.
- S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- U. Brandes, M. Gaertler, and D. Wagner. Engineering graph clustering: Models and experimental evaluation. *Journal of Experimental Algorithmics*, 12(1), 2007.
- J. Brank and J. Leskovec. The download estimation task on kdd cup 2003. *SIGKDD Explorations*, 5(2): 160–162, 2003.
- A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: experiments and models. In *WWW '00: Proceedings of the 9th international conference on World Wide Web*, 2000.
- P. Bronson. Hotmale. *Wired Magazine*, 6(12), 1998.

- J. J. Brown and P. H. Reingen. Social ties and word-of-mouth referral behavior. *The Journal of Consumer Research*, 14(3):350–362, 1987.
- E. Brynjolfsson, Y. Hu, and M. D. Smith. Consumer surplus in the digital economy: Estimating the value of increased product variety at online booksellers. *Management Science*, 49(11):1580–1596, 2003.
- S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming (series B)*, 95(2):329–357, 2003.
- C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, 2005.
- K. Burke. As consumer attitudes shift, so must marketing strategies. <http://www.audioholics.com/contact/ConsumerAttitudesMarketingStrategiesStudy.html>, 2003.
- C. T. Butts. Permutation models for relational data. (tech. rep. mbs 05-02, Univ. of California, Irvine, 2005.
- A. Capocci, V. D. P. Servedio, F. Colaiori, L. S. Buriol, D. Donato, S. Leonardi, and G. Caldarelli. Preferential attachment in the growth of social networks: The internet encyclopedia wikipedia. *Physical Review E*, 74(3), 2006.
- J. M. Carlson and J. Doyle. Highly optimized tolerance: a mechanism for power laws in designed systems. *Physical Review E*, 60(2):1412–1427, 1999.
- D. Centola and M. Macy. Complex contagion and the weakness of long ties. <ftp://hive.soc.cornell.edu/mwm14/webpage/WLT.pdf>, 2005.
- D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Computing Survey*, 38(1):2, 2006.
- D. Chakrabarti, Y. Zhan, and C. Faloutsos. R-mat: A recursive model for graph mining. In *SDM '04: SIAM Conference on Data Mining*, 2004.
- D. Chakrabarti, C. Faloutsos, and Y. Zhan. Visualization of large networks with min-cut plots, a-plots and r-mat. *Int. J. Hum.-Comput. Stud.*, 65(5):434–445, 2007a.
- D. Chakrabarti, J. Leskovec, C. Faloutsos, S. Madden, C. Guestrin, and M. Faloutsos. Information survival threshold in sensor and p2p networks. In *INFOCOM '07: Proceedings of the 26th annual IEEE Conference on Computer Communications*, pages 1316–1324, 2007b.
- D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos. Epidemic thresholds in real networks. *ACM Trans. Inf. Syst. Secur.*, 10(4):1–26, 2008.
- S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: a new approach to topic-specific web resource discovery. In *WWW '99: Proceedings of the 8th international conference on World Wide Web*, 1999.
- J. Cheeger. A lower bound for the smallest eigenvalue of the laplacian. In *Problems in Analysis, Papers dedicated to Salomon Bochner*, pages 195–199. Princeton University Press, 1969.
- B. V. Cherkassky and A. V. Goldberg. On implementing push-relabel method for the maximum flow problem. In *IPCO '95: Proceedings of the 4th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 157–171, 1995.
- J. Chevalier and D. Mayzlin. The effect of word of mouth on sales: Online book reviews. *Journal of Marketing Research*, 43(3):345, 2006.

- D. M. Chickering. The winmine toolkit. Technical Report MSR-TR-2002-103, Microsoft, 2002.
- D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2003.
- T. Chow. The Q-spectrum and spanning trees of tensor products of bipartite graphs. *Proc. Amer. Math. Soc.*, 125:3155–3161, 1997.
- F. Chung and L. Lu. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25):15879–15882, 2002a.
- F. R. K. Chung. Four proofs of cheeger inequality and graph partition algorithms. In *Proceedings of ICCM*, 2007a.
- F. R. K. Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, 2007b.
- F. R. K. Chung. Random walks and local cuts in graphs. *Linear Algebra and its Applications*, 423:22–32, 2007c.
- F. R. K. Chung. *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.
- F. R. K. Chung and L. Lu. The diameter of sparse random graphs. *Advances in Applied Mathematics*, 26(4):257–279, 2001.
- F. R. K. Chung and L. Lu. Connected components in random graphs with given expected degree sequences. *Annals of Combinatorics*, 6(2):125–145, 2002b.
- F. R. K. Chung and L. Lu. The average distances in a random graph with given expected degrees. *Internet Mathematics*, 1:91–113, 2003.
- F. R. K. Chung and L. Lu. *Complex Graphs and Networks*, volume 107 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 2006a.
- F. R. K. Chung and L. Lu. The volume of the giant component of a random graph with given expected degrees. *SIAM Journal on Discrete Mathematics*, 20:395–411, 2006b.
- F. R. K. Chung, L. Lu, and V. Vu. Eigenvalues of random power law graphs. *Annals of Combinatorics*, 7:21–33, 2003a.
- F. R. K. Chung, L. Lu, and V. Vu. The spectra of random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 100(11):6313–6318, 2003b.
- F. R. K. Chung, L. Lu, and V. Vu. The spectra of random graphs with given expected degrees. *Internet Mathematics*, 1:257–275, 2004.
- A. Clauset. Finding local community structure in networks. *Physical Review E*, 72:026132, 2005.
- A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70:066111, 2004.
- A. Clauset, C. Moore, and M. E. J. Newman. Structural inference of hierarchies in networks. *ArXiv*, physics/0610051, 2006.
- A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *ArXiv*, ArXiv:0706.1062, Jun 2007.
- A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

- R. Cohen, S. Havlin, and D. ben Avraham. Efficient immunization strategies for computer networks and populations. *Physical Review Letters*, 91:247901, 2003.
- V. Colizza, A. Flammini, M. A. Serrano, and A. Vespignani. Characterization and modeling of protein protein interaction networks. *Physica A Statistical Mechanics and its Applications*, 352:1–27, 2005.
- C. Cooper and A. Frieze. A general model of web graphs. *Random Structures and Algorithms*, 22(3):311–335, 2003.
- S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, 2002.
- M. E. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: evidence and possible causes. *IEEE /ACM Transactions on Networking*, 5(6):835–846, 1997.
- L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1):167–242, 2007.
- L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 29(09):P09008, 2005.
- D. J. de Solla Price. Networks of scientific papers. *Science*, 149(3683):510–515, July 1965.
- J. Dean and M. R. Henzinger. Finding related pages in the world wide web. In *WWW '99: Proceedings of the 8th international conference on World Wide Web*, 1999.
- A. DeBruyn and G. Lilien. A multi-stage model of word of mouth through electronic referrals. *Working paper eBusiness Research Center Paper Series*, 2004.
- I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Trans Pattern Anal Mach Intell*, 29(11):1944–1957, November 2007.
- S. Dill, R. Kumar, K. S. Mccurley, S. Rajagopalan, D. Sivakumar, and A. Tomkins. Self-similarity in the web. *ACM Trans. Interet Technology*, 2(3):205–223, 2002.
- P. Domingos and M. Richardson. Mining the network value of customers. In *KDD '01: Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001.
- W. E. Donath and A. J. Hoffman. Algorithms for partitioning graphs and computer logic based on eigenvectors of connection matrices. *IBM Technical Disclosure Bulletin*, 15(3):938–944, 1972.
- G. Dorini, P. Jonkergouw, and et al. An efficient algorithm for sensor placement in water distribution systems. In *8th Annual Symposium on Water Distribution Systems Analysis*, Cincinnati, Ohio, 2006.
- S. N. Dorogovtsev and J. F. F. Mendes. Effect of the accelerated growth of communications networks on their structure. *Phys. Rev. E*, 63(025101):025101, 2001a.
- S. N. Dorogovtsev and J. F. F. Mendes. Evolution of networks. *Advances in Physics*, 51:1079–1187, 2002a.
- S. N. Dorogovtsev and J. F. F. Mendes. Accelerated growth of networks. In *Handbook of Graphs and Networks: From the Genome to the Internet*, eds. S. Bornholdt and H. G. Schuster. Wiley-VCH, Berlin, 2002b.
- S. N. Dorogovtsev and J. F. F. Mendes. Language as an evolving word web. *Proc. Royal Soc. London B*, 268(2603):2603–2606, 2001b.
- S. N. Dorogovtsev and J. F. F. Mendes. *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford University Press, Oxford, 2003.

- S. N. Dorogovtsev, J. F. F. Mendes, and A. N. Samukhin. Structure of growing networks with preferential linking. *Phys Rev Lett*, 85(21):4633–4636, November 2000.
- S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes. Pseudofractal scale-free web. *Physical Review E*, 65(6):066122, Jun 2002.
- J. Doyle and J. M. Carlson. Power laws, highly optimized tolerance, and generalized source coding. *Physical Review Letters*, 84(24):5656–5659, 2000.
- J. Doyle and J. M. Carlson. Complexity and robustness. *Proceedings of the National Academy of Sciences*, 99:2538–2545, 2002.
- R. Dunbar. *Grooming, Gossip, and the Evolution of Language*. October 1998.
- B. Efron. Defining the curvature of a statistical problem (with applications to second order efficiency). *Ann. Statist.*, 3(6):1189–1242, 1975.
- V. M. Equiluz and K. Klemm. Epidemic threshold in structured scale-free networks. *ArXiv*, cond-mat/02055439, May 21 2002.
- P. Erdős and A. Rényi. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Academy of Science*, 5:17–67, 1960.
- G. Ergün and G. J. Rodgers. Growing random networks with fitness. *Physica A: Statistical Mechanics and its Applications*, 303(1-2):261–272, Jan 2002.
- E. Estrada. Spectral scaling and good expansion properties in complex networks. *Europhysics Letters*, 73: 649–655, 2006.
- A. Fabrikant, E. Koutsoupias, and C. H. Papadimitriou. Heuristically optimized trade-offs: A new paradigm for power laws in the internet. In *ICALP '02: Proceedings of the 29th International Colloquium on Automata, Languages, and Programming*, volume 2380, 2002.
- M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pages 251–262, 1999.
- I. Farkas, I. Deréni, A.-L. Barabási, and T. Vicsek. Spectra of “real-world” graphs: beyond the semicircle law. *Physical Review E*, 64(026704), 2001.
- U. Feige and E. Ofek. Spectral techniques applied to sparse random graphs. *Random Structures and Algorithms*, 27:251–275, 2005.
- D. Fernholz and V. Ramachandran. The diameter of sparse random graphs. *Random Structures and Algorithms*, 31:482–516, 2007.
- D. Fetterly, M. Manasse, M. Najork, and J. Wiener. A large-scale study of the evolution of web pages. *Software Practice and Experience*, 34(2):213–237, 2004.
- C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *DAC '82: Proceedings of the 19th ACM/IEEE Conference on Design Automation*, pages 175–181, 1982.
- M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Math. Journal*, 23(98):298–305, 1973.
- G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–160, 2000.
- G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee. Self-organization and identification of web communities. *IEEE Computer*, 35(3):66–71, 2002.

- G. W. Flake, R. E. Tarjan, and K. Tsoutsouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 1(4):385–408, 2003.
- A. D. Flaxman, A. M. Frieze, and J. Vera. A geometric preferential attachment model of networks. In *WAW '04: Proceedings of the 2nd Workshop On Algorithms And Models For The Web-Graph*, pages 44–55, 2004.
- A. D. Flaxman, A. Frieze, and T. Fenner. High degree vertices and eigenvalues in the preferential attachment graph. *Internet Mathematics*, 2(1):1–19, 2005.
- A. D. Flaxman, A. M. Frieze, and J. Vera. A geometric preferential attachment model of networks II. In *WAW '07: Proceedings of the 5th Workshop On Algorithms And Models For The Web-Graph*, pages 41–55, 2007.
- S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.
- N. Fountoulakis and B. Reed. The evolution of the mixing rate. *ArXiv*, math/0701474, January 2007a.
- N. Fountoulakis and B. A. Reed. Faster mixing and small bottlenecks. *Probability Theory and Related Fields*, 137(3–4):475–486, 2007b.
- J. Frenzen and K. Nakamoto. Structure, cooperation, and the flow of market information. *Journal of Consumer Research*, 20(3):360–375, December 1993.
- M. Gaertler. Clustering. In U. Brandes and T. Erlebach, editors, *Network Analysis: Methodological Foundations*, pages 178–215. Springer, 2005.
- G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55, 1989.
- D. Gamerman. *Markov chain Monte Carlo, Stochastic simulation for Bayesian inference*. Chapman & Hall, London, 1997.
- E. Garfield. Citation analysis as a tool in journal evaluation. *Science*, 178(60):471–479, November 1972.
- J. Gehrke, P. Ginsparg, and J. M. Kleinberg. Overview of the 2003 kdd cup. *SIGKDD Explorations*, 5(2):149–151, 2003.
- A. Gelman, J. B. Carlin, H. S. Stern, and D.B. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall, London, July 2003.
- G. Giakkoupis, A. Gionis, E. Terzi, and P. Tsaparas. Models and algorithms for network immunization. Technical Report C-2005-75, Department of Computer Science, University of Helsinki, 2005.
- R. Gibrat. Les inégalités Économiques. *Paris: Librairie du Recueil Sirey*, 1931.
- D. Gibson, J. M. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *HT '98: Proceedings of the 9th ACM Conference on Hypertext and Hypermedia*, pages 225–234, 1998.
- M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002.
- C. Gkantsidis, M. Mihail, and A. Saberi. Conductance and congestion in power law graphs. In *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and modeling of computer systems*, pages 148–159, 2003.
- N. Glance, M. Hurst, K. Nigam, M. Siegler, R. Stockton, and T. Tomokiyo. Deriving marketing intelligence from online discussion. In *KDD '05: Proceeding of the 11th ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 419–428, 2005.

- A. V. Goldberg and S. Rao. Beyond the flow decomposition barrier. *Journal of the ACM*, 45:783–797, 1998.
- A. V. Goldberg and R. E. Tarjan. A new approach to the maximum-flow problem. *Journal of the ACM*, 35:921–940, 1988.
- J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 3(12):211–223, 2001.
- L. Gomes. It may be a long time before the long tail is wagging the web. *The Wall Street Journal*, 2006. July 26 2006.
- Google. Google Programming Contest. <http://www.google.com/programming-contest/>, 2002.
- M. S. Granovetter. The strength of weak ties. *American Journal of Sociology*, 78:1360–1380, 1973.
- M. S. Granovetter. Threshold models of collective behavior. *American Journal of Sociology*, 83(6):1420–1443, 1978.
- D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 491–501, 2004.
- J. Guan, M. M. Aral, and et al. Optimization model and algorithms for design of water sensor placement in water distribution systems. In *8th Annual Symposium on Water Distribution Systems Analysis*, Cincinnati, Ohio, 2006.
- S. Guattery and G. L. Miller. On the quality of spectral separators. *SIAM Journal on Matrix Analysis and Applications*, 19:701–719, 1998.
- R. Gueli. Predator-prey model for discrete sensor placement. In *8th Annual Symposium on Water Distribution Systems Analysis*, Cincinnati, Ohio, 2006.
- R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral. Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70:025101, 2004.
- B. H. Hall, A. B. Jaffe, and M. Trajtenberg. The nber patent citation data file: Lessons, insights and methodological tools. NBER Working Papers 8498, National Bureau of Economic Research, Inc, October 2001.
- M. B. Hastings. Community detection as an inference problem. *Physical Review E*, 74:035102, 2006.
- T. H. Haveliwala. Topic-sensitive pagerank. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, 2002.
- B. Hendrickson and R. Leland. A multilevel algorithm for partitioning graphs. In *Supercomputing '95: Proceedings of the 1995 ACM/IEEE Conference on Supercomputing (CDROM)*, 1995.
- H. W. Hethcote. The mathematics of infectious diseases. *SIAM Rev.*, 42(4):599–653, 2000.
- S. Hill, F. Provost, and C. Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, 21(2):256–276, 2006.
- P. Holme. Core-periphery organization of complex networks. *Physical Review E*, 72:046111, 2005.
- P. Holme and M. E. J. Newman. Nonequilibrium phase transition in the coevolution of networks and opinions. *Physical Review E*, 74:056108, 2006.
- S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43:439–561, 2006.

- J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Natural communities in large linked networks. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 541–546, 2003.
- J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. *Proceedings of the National Academy of Sciences*, 101:5249–5253, 2004.
- B. A. Huberman and L. A. Adamic. Growth dynamics of the world-wide web. *Nature*, 399:131, 1999.
- IDC Market Analysis. Worldwide enterprise instant messaging applications 2005–2009 forecast and 2004 vendor shares: Clearing the decks for substantial growth, 2005.
- A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31:264–323, 1999.
- G. Jeh and J. Widom. Scaling personalized web search. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, 2003.
- H. Jeong, S. P. Mason, A.-L. Barabási, and Z. N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411:41–42, 2001.
- N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous univariate distributions: Vol. 1, 2nd Edition*. John Wiley, New York, 1994.
- R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, 2006.
- S. Jurvetson. What exactly is viral marketing? *Red Herring*, 78:110–112, 2000.
- R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515, 2004.
- R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
- B. Karrer, E. Levina, and M. E. J. Newman. Robustness of community structure in networks. *Physical Review E*, 77:046119, 2008.
- G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *J. Parallel Distrib. Comput.*, 48:96–129, 1998a.
- G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1998b.
- H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *ICML '03: Proceedings of the 20th international conference on Machine learning*, 2003.
- J. S. Katz. Scale independent bibliometric indicators. *Measurement: Interdisciplinary Research and Perspectives*, 3:24–28, 2005.
- J. S. Katz. The self-similar science system. *Research Policy*, 28:501–517, 1999.
- D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD '03: Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, 2003.
- J. Kepner. Kronecker theory of power law graphs. In *SIAM PP '08: Workshop for HPC on Large Graphs*, 2008.
- B. Kernighan and S. Lin. An effective heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, pages 291–308, 1970.

- A. Khalil and Y. Liu. Experiments with PageRank computation. Technical Report 603, Indiana University Department of Computer Science, December 2004.
- R. Khandekar, S. Rao, and U. Vazirani. Graph partitioning using single commodity flows. In *STOC '06: Proceedings of the 38th annual ACM Symposium on Theory of Computing*, pages 385–390, 2006.
- S. Khuller, A. Moss, and J. Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45, 1999.
- P. Killworth and H. Bernard. Reverse small world experiment. *Social Networks*, 1:159–192, 1978.
- J. M. Kleinberg. Small-world phenomena and the dynamics of information. In *NIPS '02: Advances in Neural Information Processing Systems 14*, 2002.
- J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999a.
- J. M. Kleinberg. The small-world phenomenon: an algorithmic perspective. Technical Report 99-1776, Cornell Computer Science Department, 1999b.
- J. M. Kleinberg, S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. The web as a graph: Measurements, models and methods. In *COCOON '99: Proceedings of the International Conference on Combinatorics and Computing*, 1999.
- B. Klimt and Y. Yang. Introducing the enron corpus. In *CEAS '04: Proceedings of the 1st Conference on Email and Anti-Spam*, pages 1–2, 2004.
- R. I. Kondor and J. D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML '02: Proceedings of the 19th international conference on Machine learning*, 2002.
- G. Kossinets and D. J. Watts. Empirical analysis of an evolving social network. *Science*, 311(5757):88–90, January 2006.
- D. Krackhardt and M. Handcock. Heider vs. Simmel: Emergent features in dynamic structure. In *Statistical Network Analysis: Models, Issues, and New Directions*, pages 14–27, 2007.
- P. L. Krapivsky and S. Redner. Organization of growing random networks. *Phys. Rev. E*, 63(6):066123, May 2001.
- P. L. Krapivsky and S. Redner. Network growth by copying. *Physical Review E*, 71(036118):036118, 2005.
- A. Krause and C. Guestrin. A note on the budgeted maximization of submodular functions. Technical Report CMU-CALD-05-103, Carnegie Mellon University, 2005.
- A. Krause, J. Leskovec, C. Guestrin, J. VanBriesen, and C. Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *Accepted to the Journal of Water Resources Planning an Management*, 0:0, 2008.
- T. Kudo, E. Maeda, and Y. Matsumoto. An application of boosting to graph classification. In *NIPS '04: Advances in Neural Information Processing Systems 17*, 2004.
- S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Math. Stat.*, 22(1):79–86, 1951.
- R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 57, 2000.
- R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. On the bursty evolution of blogspace. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 568–576, 2003.

- R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 611–617, 2006.
- S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Extracting large-scale knowledge bases from the web. In *Proceedings of the 25th VLDB Conference*, Edinburgh, Scotland, 1999a.
- S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493, 1999b.
- K. Lang. Finding good nearly balanced cuts in power law graphs. Technical Report YRL-2004-036, Yahoo! Research Labs, Pasadena, CA, November 2004.
- K. Lang and S. Rao. A flow-based method for improving the expansion or conductance of graph cuts. In *IPCO '04: Proceedings of the 10th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 325–337, 2004.
- K. Lang and S. Rao. Finding near-optimal cuts: an empirical evaluation. In *SODA '93: Proceedings of the 4th annual ACM-SIAM Symposium on Discrete algorithms*, pages 212–221, 1993.
- A. N. Langville and W. J. Stewart. The Kronecker product and stochastic automata networks. *Journal of Computation and Applied Mathematics*, 167:429–447, 2004.
- T. Lau and E. Horvitz. Patterns of search: analyzing and modeling web query refinement. In *UM '99: Proceedings of the seventh international conference on User modeling*, 1999.
- T. Leighton and S. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *FOCS '88: Proceedings of the 28th Annual Symposium on Foundations of Computer Science*, pages 422–431, 1988.
- T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, 1999.
- J. Leskovec and C. Faloutsos. Sampling from large graphs. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636, 2006.
- J. Leskovec and C. Faloutsos. Scalable modeling of real graphs using kronecker multiplication. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, 2007.
- J. Leskovec and E. Horvitz. Planetary-scale views on a large instant-messaging network. In *WWW '08: Proceedings of the 17th International Conference on World Wide Web*, 2008.
- J. Leskovec, D. Chakrabarti, J. M. Kleinberg, and C. Faloutsos. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In *PKDD '05: Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 133–145, 2005a.
- J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD '05: Proceeding of the 11th ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187, 2005b.
- J. Leskovec, N. Milic-Frayling, and M. Grobelnik. Impact of linguistic analysis on the semantic graph coverage and learning of document extracts. In *AAAI '05: Proceedings of the 20th National Conference on Artificial Intelligence*, pages 1069–1074, 2005c.
- J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. In *EC '06: Proceedings of the 7th ACM conference on Electronic commerce*, pages 228–237, 2006a.

- J. Leskovec, A. Singh, and J. M. Kleinberg. Patterns of influence in a recommendation network. In *PAKDD '06: Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 380–389, 2006b.
- J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):2, 2007a.
- J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, 2007b.
- J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD '07: Proceeding of the 13th ACM SIGKDD international conference on Knowledge discovery in data mining*, 2007c.
- J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs. In *SDM '07: Proceedings of the SIAM Conference on Data Mining*, 2007d.
- J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *KDD '08: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470, 2008a.
- J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW '08: Proceedings of the 17th International Conference on World Wide Web*, 2008b.
- J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *ArXiv*, arXiv:0810.1355, Oct 2008c.
- L. Li, D. Alderson, John C. Doyle, and W. Willinger. Towards a theory of scale-free graphs: Definition, properties, and implications. *Internet Mathematics*, 2(4):431–523, 2005.
- D. Liben-Nowell and J. M. Kleinberg. The link prediction problem for social networks. In *12th CIKM*, pages 556–559, 2003.
- D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geographic routing in social networks. *Proceedings of the National Academy of Sciences*, 102(33):11623–11628, 2005.
- G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.
- C. F. Van Loan. The ubiquitous Kronecker product. *Journal of Computation and Applied Mathematics*, 123:85–100, 2000.
- M. O. Lorenz. Methods of measuring the concentration of wealth. *Publications of the American Statistical Association*, 9(70):209–219, 1905.
- L. Lu. The diameter of random massive graphs. In *SODA '01: Proceedings of the 12th annual ACM-SIAM Symposium on Discrete algorithms*, pages 912–921, 2001.
- D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54:396–405, 2003.
- M. Mahdian and Y. Xu. Stochastic kronecker graphs. In *WAW '07: Proceedings of the 5th Workshop On Algorithms And Models For The Web-Graph*, pages 179–186, 2007.
- P. V. Marsden. Core discussion networks of americans. *American Sociological Review*, 52(1):122–131, 1987.

- M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.
- F. Menczer. Growing and navigating the small world web by local content. *Proceedings of the National Academy of Sciences*, 99(22):14014–14019, 2002.
- M. Mihail and C. H. Papadimitriou. On the eigenvalue power law. In *RANDOM '02: Proceedings of the 6th International Workshop on Randomization and Approximation Techniques*, pages 254–262, 2002.
- M. Mihail, C.H. Papadimitriou, and A. Saberi. On certain connectivity properties of the internet topology. *Journal of Computer and System Sciences*, 72(2):239–251, 2006.
- S. Milgram. The small-world problem. *Psychology Today*, 2:60–67, 1967.
- R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network Motifs: Simple Building Blocks of Complex Networks. *Science*, 298(5594):824–827, 2002.
- R. Milo, N. Kashtan, S. Itzkovitz, M. E. J. Newman, and U. Alon. On the uniform generation of random graphs with prescribed degree sequences. *ArXiv*, cond-mat/0312028, May 2004.
- E. Minkov, W. W. Cohen, and A. Y. Ng. Contextual search and name disambiguation in email using graphs. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.
- M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1(2):226–251, 2004.
- B. Mohar. The Laplacian spectrum of graphs. In Y. Alavi, G. Chartrand, O. R. Oellermann, and A. J. Schwenk, editors, *Graph Theory, Combinatorics, and Applications, Vol. 2*, pages 871–898. Wiley, 1991.
- M. Molloy and B. Reed. A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms*, 6:161–180, 1995.
- M. Molloy and B. Reed. The size of the giant component of a random graph with a given degree sequence. *Combinatorics, Probability and Computing*, 7:295–305, 1998.
- A. L. Montgomery. Applying quantitative marketing techniques to the internet. *Interfaces*, 30:90–108, 2001.
- A. L. Montgomery and C. Faloutsos. Identifying web browsing trends and patterns. *IEEE Computer*, 34(7):94–95, 2001.
- B. A. Nardi, S. Whittaker, and E. Bradner. Interaction and outeraction: instant messaging in action. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 79–88, 2000.
- G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.
- G. L. Nemhauser and L. A. Wolsey. *Studies on Graphs and Discrete Programming*, chapter Maximizing Submodular Set Functions: Formulations and Analysis of Algorithms, pages 279–301. North-Holland, 1981.
- Netflix. Netflix prize. <http://www.netflixprize.com>, 2006.
- Network data. Collection of network datasets, mainly small networks used for community detection. <http://www-personal.umich.edu/~mejn/netdata>, 2007.
- M. E. J. Newman. Clustering and preferential attachment in growing networks. *Physical Review E*, 64:025102, 2001.

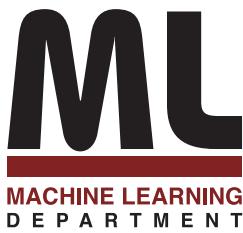
- M. E. J. Newman. The spread of epidemic disease on networks. *Physical Review E*, 66:016128, 2002.
- M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.
- M. E. J. Newman. Detecting community structure in networks. *The European Physical Journal B*, 38: 321–330, 2004.
- M. E. J. Newman. Power laws, pareto distributions and zipf’s law. *Contemporary Physics*, 46:323–351, December 2005.
- M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74:036104, 2006a.
- M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006b.
- M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, 2004.
- M. E. J. Newman and E. A. Leicht. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences*, 104(23):9564–9569, 2007.
- M. E. J. Newman, S. Forrest, and J. Balthrop. Email networks and the spread of computer viruses. *Physical Review E*, 66(3):035101, 2002.
- C. L. M. Nickel. Random dot product graphs: A model for social networks. Ph.D. Thesis, Dept. of Applied Mathematics and Statistics, Johns Hopkins University, 2008.
- L. Nie, B. D. Davison, and X. Qi. Topical link analysis for web search. In *SIGIR ’06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.
- Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma. Object-level ranking: bringing order to web objects. In *WWW ’05: Proceedings of the 14th international conference on World Wide Web*, 2005.
- NIST. TREC Web Corpus: WT10g. http://ir.dcs.gla.ac.uk/test_collections/wt10g.html, 2000.
- A. Ntoutas, J. Cho, and C. Olston. What’s new on the web? the evolution of the web from a search engine perspective. In *WWW ’04: Proceedings of the 13th international conference on World Wide Web*, pages 1–12, 2004.
- T. O’Reilly. O’reilly network: What is web 2.0, September 2005.
- A. Ostfeld and E. Salomons. Optimal layout of early warning detection stations for water distribution systems security. *J. Water Resources Planning and Management*, 130(5):377–385, 2004.
- A. Ostfeld, J. G. Uber, and E. Salomons. Battle of water sensor networks: A design challenge for engineers and algorithms. In *8th Symposium on Water Distribution Systems Analysis*, 2006.
- L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Dig. Lib. Tech. Proj., 1998.
- G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.
- C. R. Palmer, P. B. Gibbons, and C. Faloutsos. Anf: a fast and scalable tool for data mining in massive graphs. In *KDD ’02: Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 81–90, 2002.
- R. Pastor-Satorras and A. Vespignani. Immunization of complex networks. *Physical Review E*, 65(3): 036104, 2002.

- D. M. Pennock, G. W. Flake, S. Lawrence, E. J. Glover, and C. L. Giles. Winners don't take all: Characterizing the competition for links on the Web. *Proceedings of the National Academy of Sciences*, 99(8):5207–5211, 2002.
- G. Pinski and F. Narin. Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics. *Information Processing and Management*, 12:297–312, 1976.
- Y. Qi, J. Klein-Seetharaman, and Z. Bar-Joseph. Random forest similarity for protein-protein interaction prediction from multiple sources. In *Pacific Symposium on Biocomputing*, pages 531–542, 2005.
- F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, 101(9):2658–2663, 2004.
- F. Radlinski and S. Dumais. Improving personalized web search using result diversification. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.
- U. Nandini Raghavan, R. Z. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76:036106, 2007.
- E. Ravasz and A.-L. Barabási. Hierarchical organization in complex networks. *Physical Review E*, 67(2):026112, 2003.
- E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555, 2002.
- S. Redner. Citation statistics from more than a century of physical review. *ArXiv*, physics/0407137, 2004.
- S. Redner. How popular is your paper? an empirical study of the citation distribution. *European Physical Journal B*, 4:131–134, 1998.
- J. Reichardt and S. Bornholdt. Partitioning and modularity of graphs with arbitrary degree distribution. *Physical Review E*, 76:015102, 2007.
- Y. Ren, R. Kraut, and S. Kiesler. Applying common identity and bond theory to design of online communities. *Organization Studies*, 28(3):377–408, 2007.
- P. Resnick and R. Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. In *The Economics of the Internet and E-Commerce*. 2002.
- M. Richardson and P. Domingos. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *NIPS '02: Advances in Neural Information Processing Systems 14*, 2002a.
- M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD '02: Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 61–70, 2002b.
- M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *ISWC '03: Proceedings of the 2nd International Semantic Web Conference*, 2003.
- M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, 6(1):50–57, Jan/Feb 2002.
- J. Rissanen. Modelling by the shortest data description. *Automatica*, 14:465–471, 1978.
- T. G. Robertazzi and S. C. Schwartz. An accelerated sequential algorithm for producing D-optimal designs. *SIAM J. of Scientific and Statistical Computing*, 10(2):341–358, March 1989.
- E. M. Rogers. *Diffusion of Innovations*. New York, fourth edition, 1995.

- E. M. Rogers and D. K. Bhowmik. Homophily-heterophily: Relational concepts for communication research. *Public Opinion Quarterly*, 34:523–538, 1970.
- L. A. Rossman. The epanet programmer’s toolkit for analysis of water distribution systems. In *Annual Water Resources Planning and Management Conference*, 1999.
- M. Rosvall and C. T. Bergstrom. An information-theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, 104(18):7327–7331, 2007.
- RouteViews. University of Oregon Route Views Project. Online data and reports. <http://www.routeviews.org>, 1997.
- M. Sales-Pardo, R. Guimera, A. A. Moreira, and L. A. Amaral. Extracting the hierarchical organization of complex systems. *Proceedings of the National Academy of Sciences*, 104(39):15224–15229, September 2007.
- S. F. Sampson. A novitiate in a period of change: An experimental and case study of social relationships. Ph.D. Thesis, Dept. of Sociology, Cornell Univ., 1968.
- S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.
- M. Schroeder. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. W. H. Freeman and Company, New York, 1991.
- G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.
- S. S. S. Shai, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nature Genetics*, 31:64–68, 2002.
- J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- X. Shi, L. A. Adamic, and M. J. Strauss. Networks of strong ties. *Physica A*, 378(1):3347, 2007.
- G. Siganos, S. L. Tauro, and M. Faloutsos. Jellyfish: A conceptual model for the as internet topology. *Journal of Communications and Networks*, 8:339–350, 2006.
- H. A. Simon. On a class of skew distribution functions. *Biometrika*, 42:425–440, 1955.
- P. Singla and M. Richardson. Yes, there is a correlation from social networks to personal behavior on the web. In *WWW ’08: Proceedings of the 17th international conference on World Wide Web*, 2008.
- R. Sole and B. Goodwin. *Signs of Life: How Complexity Pervades Biology*. Perseus Books Group, New York, NY, 2000.
- D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *STOC ’04: Proceedings of the 36th annual ACM Symposium on Theory of Computing*, pages 81–90, 2004.
- D. A. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *FOCS ’96: Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, pages 96–107, 1996.
- D. Strang and S. A. Soule. Diffusion in organizations and social movements: From hybrid corn to poison pills. *Annual Review of Sociology*, 24:265–290, 1998.
- S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.
- M. P. Stumpf, C. Wiuf, and R. M. May. Subnets of scale-free networks are not scale-free: sampling properties of networks. *Proceedings of the National Academy of Sciences*, 102(12):4221–4224, March 2005.

- M. R. Subramani and B. Rajagopalan. Knowledge-sharing and influence in online social networks via viral marketing. *Communications of the ACM*, 46(12):300–307, 2003.
- M. Sviridenko. A note on maximizing a submodular set function subject to knapsack constraint. *Operations Research Letters*, 32:41–43, 2004.
- S. L. Tauro, C. Palmer, G. Siganos, and M. Faloutsos. A simple conceptual model for the internet topology. In *GLOBECOM '01: Global Telecommunications Conference*, volume 3, pages 1667 – 1671, 2001.
- J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- J. Travers and S. Milgram. An experimental study of the small world problem. *Sociometry*, 32:425–443, 1969.
- C. E. Tsourakakis. Fast counting of triangles in real-world networks: proofs, algorithms and observations. Technical report cmu-ml-08-103, Carnegie Mellon University, May 2008.
- S. Vassilvitskii and E. Brill. Using web-graph distance for relevance feedback in web search. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.
- A. Vazquez. Disordered networks generated by recursive searches. *Europhysics Letters*, 54(4):430–435, 2001.
- A. Vazquez. Growing networks with local rules: preferential attachment, clustering hierarchy and degree correlations. *Physical Review E*, 67(5):056104, 2003.
- A. Vazquez, J. G. Oliveira, Z. Dezso, K.-I. Goh, I. Kondor, and A.-L. Barabási. Modeling bursts and heavy tails in human dynamics. *Physical Review E*, 73(3):036127, 2006.
- A. Voida, W. C. Newstetter, and E. D. Mynatt. When conventions collide: the tensions of instant messaging attributed. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 187–194, 2002.
- U. von Luxburg. A tutorial on spectral clustering. Technical Report 149, Max Plank Institute for Biological Cybernetics, August 2006.
- M. Wang, T. Madhyastha, N. H. Chang, S. Papadimitriou, and C. Faloutsos. Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic. In *ICDE*, 2002.
- Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. In *SRDS*, pages 25–34, 2003.
- S. Wasserman and K. Faust. *Social Network Analysis : Methods and Applications*. November 1994.
- S. Wasserman and P. Pattison. Logit models and logistic regressions for social networks. *Psychometrika*, 60:401–425, 1996.
- D. J. Watts. A simple model of global cascades on random networks. *Proceedings of the National Academy of Sciences*, 99(9):4766–5771, Apr 2002.
- D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.
- D. J. Watts, P. S. Dodds, and M. E. J. Newman. Identity and search in social networks. *Science*, 296: 1302–1305, 2002.
- B. M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, December 1988.

- S. White and P. Smyth. A spectral clustering approach to finding communities in graphs. In *SDM '05: Proceedings of the 5th SIAM International Conference on Data Mining*, pages 76–84, 2005.
- J. Winick and S. Jamin. Inet-3.0: Internet Topology Generator. Technical Report CSE-TR-456-02, University of Michigan, Ann Arbor, 2002.
- C. Wiuf, M. Brämeier, O. Hagberg, and M. P. Stumpf. A likelihood approach to analysis of network data. *Proceedings of the National Academy of Sciences*, 103(20):7566–7570, 2006.
- B. Wu, V. Goel, and B. D. Davison. Topical trustrank: using topicality to combat web spam. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, 2006.
- F. Wu and B. A. Huberman. Finding communities in linear time: a physics approach. *The European Physical Journal B*, 38(2):331–338, 2004a.
- F. Wu and B. A. Huberman. Social structure and opinion formation. *Computational Economics* 0407002, EconWPA, Jul 2004b. Available at <http://ideas.repec.org/p/wpa/wuwpc0/0407002.html>.
- W. Xi, B. Zhang, Z. Chen, Y. Lu, S. Yan, W.-Y. Ma, and E. A. Fox. Link fusion: a unified link analysis framework for multi-type interrelated data objects. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 2004.
- Z. Xiao, L. Guo, and J. Tracey. Understanding instant messaging traffic characteristics. In *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*, page 51, 2007.
- Q. Xuan, Y. Li, and T.-J. Wu. Growth model for complex networks with hierarchical and modular structures. *Physical Review E*, 73:036105, 2006.
- S. Yang and G. M. Allenby. Modeling interdependent consumer preferences. *Journal of Marketing Research*, 40(3):282 – 294, August 2003.
- S. J. Young and E. R. Scheinerman. Random dot product graph models for social networks. In *WAW '07: Proceedings of the 5nd Workshop On Algorithms And Models For The Web-Graph*, pages 138–149, 2007.
- W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.
- C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, C-20(1):68–86, 1971.
- Yin Zhang, Lee Breslau, Vern Paxson, and Scott Shenker. On the characteristics and origins of internet flow rates. In *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 309–322, 2002.
- Y. Zhao and G. Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55:311–331, 2004.
- G. K. Zipf. *Human Behavior and Principle of Least Effort: An Introduction to Human Ecology*. Cambridge, Massachusetts, 1949.



Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Carnegie Mellon.

Carnegie Mellon University does not discriminate and Carnegie Mellon University is required not to discriminate in admission, employment, or administration of its programs or activities on the basis of race, color, national origin, sex or handicap in violation of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972 and Section 504 of the Rehabilitation Act of 1973 or other federal, state, or local laws or executive orders.

In addition, Carnegie Mellon University does not discriminate in admission, employment or administration of its programs on the basis of religion, creed, ancestry, belief, age, veteran status, sexual orientation or in violation of federal, state, or local laws or executive orders. However, in the judgment of the Carnegie Mellon Human Relations Commission, the Department of Defense policy of, "Don't ask, don't tell, don't pursue," excludes openly gay, lesbian and bisexual students from receiving ROTC scholarships or serving in the military. Nevertheless, all ROTC classes at Carnegie Mellon University are available to all students.

Inquiries concerning application of these statements should be directed to the Provost, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213, telephone (412) 268-6684 or the Vice President for Enrollment, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213, telephone (412) 268-2056

Obtain general information about Carnegie Mellon University by calling (412) 268-2000