

Understanding and Managing Propagation on Large Networks—Theory, Algorithms, and Models

B. Aditya Prakash

CMU-CS-12-138

September 2012

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Christos Faloutsos, Chair

Roni Rosenfeld

David Andersen

Jon Kleinberg, Cornell University, Ithaca

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

Copyright © 2012 **B. Aditya Prakash**

This research was sponsored by ICAST, the National Science Foundation under grant numbers CNS-0721736 and IIS-1017415, the Department of Energy/National Security Agency under grant number DE-AC52-07NA27344, and the Army under grant number W911NF-08-R-0013.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: data mining, graph mining, time series analysis, arbitrary networks, virus propagation, cascades, viral marketing, contagion, memes, immunization, culprits, epidemic threshold, tipping-points, winner-takes-all, co-existence, eigen-drop, information diffusion models, competing species, epidemiology, outliers, eigenvalues, NetShield, NetSleuth, Smart-Alloc, SpikeM, CTM

ॐ

न चोरहार्यम् न च राजहार्यम्
न भ्रातृभाज्यम् न च भारकारी ।
व्यये कृते वर्धत एव नित्यम्
विद्याधनम् सर्वधनप्रधानम् ॥

— A Sanskrit Verse

to
Mamma & Pappa
with love

Abstract

How do contagions spread in population networks? What happens if the networks change with time? Which hospitals should we give vaccines to, for maximum effect? How to detect sources of rumors on Twitter/Facebook? These questions and many others such as which group should we market to, for maximizing product penetration, how quickly news travels in online media and how the relative frequencies of competing tasks evolve are all related to propagation/cascade-like phenomena on networks.

In this thesis, we present novel theory, algorithms and models for propagation processes on large static and dynamic networks, focusing on:

1. **Theory:** We tackle several fundamental questions like determining if there will be an epidemic, given the underlying networks and virus propagation models and predicting who-wins when viruses (or memes or products etc.) compete. We give a unifying answer for the threshold based on eigenvalues, and prove the surprising winner-takes-all' result and other subtle phase-transitions for competition among viruses.
2. **Algorithms:** Based on our analysis, we give dramatically better algorithms for important tasks like effective immunization and reliably detecting culprits of epidemics. Thanks to our carefully designed algorithms, we achieve $6x$ fewer infections on real hospital patient-transfer graphs while also being significantly faster than other competitors (upto $30,000x$).
3. **Models:** Finally using our insights, we study numerous datasets to develop powerful general models for information diffusion and competing species in a variety of situations. Our models unify earlier patterns and results, yet being succinct and enable challenging tasks like trend forecasting, spotting outliers and answering 'what-if' questions.

Our *inter-disciplinary* approach has led to many discoveries in this thesis, with broad applications spanning areas like public health, social media, product marketing and networking. We are arguably the *first* to present a systematic study of propagation and immunization of *single* as well as *multiple* viruses on *arbitrary, real* and *time-varying* networks as the vast majority of the literature focuses on structured topologies, cliques, and related un-realistic models.

Acknowledgements

First, I want to thank Christos Faloutsos, for being a terrific advisor, mentor and friend. This thesis would not have been possible (or fun) without his generosity of time, keen insights and encouragement. I am also grateful for the academic freedom he gave me to explore challenging problems and the gentle nudges in the right direction when needed. Before joining Carnegie Mellon University, I was told that Christos is one of the nicest and most cheerful persons around—and over all these years, I have only been amazed at his ability to remain effortlessly so.

I would also like to thank my thesis committee members Roni Rosenfeld, David Andersen and Jon Kleinberg for their timely and valuable feedback and also advice during my job-search. Getting suggestions from their different perspectives has greatly improved the thesis.

In addition to Christos, I had an awesome set of collaborators and co-authors, each of whom deserves my thanks: Lada Adamic, David Andersen, Alex Beutel, Deepayan Chakrabarti, Tina Eliassi-Rad, Michalis Faloutsos, Varun Gupta, Theodore Iwashnya, Danai Koutra, Lei Li, Sridhar Machiraju, Yasuko Matsubara, Iulian Neamtiu, Kunal Punera, Roni Rosenfeld, Yasushi Sakurai, Mukund Seshadri, Ashwin Sridharan, Jack Stokes, Hanghang Tong, Charalampos Tsourakakis, Nicholas Valler, Jilles Vreeken, Xue-tao Wei and Alice Zheng. Thanks to Lada for hosting me at UMichigan, giving me thoughtful viewpoints and thinking that I am ‘too young’ to graduate! Thanks to Roni, for helping me get acquainted with the exciting world of epidemiology and ecology. Thanks to Michalis for being so enthusiastic and supportive throughout my Ph.D. Thanks to Lei and Hanghang, for many enjoyable sessions of brain-storming, when I was still a junior student.

I would also like to thank all the members and visitors of the Database Group for interesting discussions, fun trips and helpful feedback: Leman Akoglu, Alex Beutel, Rishy Chandy, Polo Chau, Robson L. F. Cordeiro, Fan Guo, U Kang, Sang-Wook Kim, Danai Koutra, Lei Li, Yasuko Matsubara, Mary McGlohon, Ippokratis Pandis, Evangelos Papalexakis, Yasushi Sakurai, Hanghang Tong, Pedro Olmo Vaz de Melo and Jilles Vreeken.

I learnt a lot during my various internships during the Ph.D.: thanks to Sridhar Machiraju, Mukund Seshadri and Ashwin Sridharan (Sprint Research Labs-Burlingame), Jack Stokes and Alice Zheng (Microsoft Research-Redmond), and Ravi Kumar, Deepayan Chakrabarti and Kunal Punera (Yahoo! Research-Santa Clara) for hosting me and collaborating on fun and interesting problems.

The Computer Science Department at Carnegie Mellon is an incredible place for graduate students. Apart from the fantastic professors and peers, the collegial and supportive atmosphere it provides to students is priceless. Thanks to Deborah Cavlovich, Catherine Copetas, Karen Lindenfelser, Marilyn Walgora and Charlotte Yano, for being so efficient and always ‘on top of things’ from countless travel re-imburements to important reminders, and making my Ph.D. experience all the more frictionless.

Thanks to all my friends at CMU, for all the help and memorable times we shared in Pittsburgh. While there are too many to list everyone here (and apologies for not attempting to do so), particular thanks to: Debabrata Dash for helping me with the transition from India to the US; Debashis Kar, Kaushik Lakshminarayanan and Satyajeet Ojha for being cool housemates; Vivek Seshadri for being my squash partner; Varun Gupta, Ravishankar Krishnaswamy and Vyas Sekar for their advice especially during my job-search; Hetunandan Kamisetty for teaching me how to drive a car; Srivatsan Narayanan and Dafna Shahaf for helping review my papers and Pranjal Awasthi and Ali Kemal Sinop for frequently joining me for Japanese desserts. Thanks also to all my officemates throughout the years—especially Sumit Jha, Kanat Tangwongsan, and Ekaterina Taralova—for making my hours at work enjoyable.

I would also like to thank all my professors and teachers from my undergraduate and schooling days in India (at the Indian Institute of Technology-Bombay and Delhi Public School-Bhilai), for providing me with an excellent foundational education. The summer internships during my time as an undergraduate, with Jayant Haritsa (IISc.-Bangalore), and Laks Lakshmanan and Raymond Ng (UBC-Vancouver), and my senior year thesis with S. Sudarshan (IIT-Bombay) were instrumental in fostering my enthusiasm for research. Additionally, thanks to Praveen Mone, for introducing me to the thrills and mysterious rhythms of the Tabla.

Above all, I wish to thank my family for being there for me through both good and difficult times: Thank you to my elder brother Kartik for setting me an example through his efforts and his advice and also my sister-in-law Shilpa for her cheerful support. A very special thanks to my four-legged furry friend, Munnu, for his affectionate woofs, and agreeing to always jump in joy at seeing me. I owe my deepest gratitude to my parents Prema and Sunder Ram Prakash, for their endless love, sacrifice, prayers and encouragement and for giving me the strength and the belief to complete this Ph.D. They were also the first to instill in me a respect for learning and led the way in nurturing a motivation and curiosity to pursue science. This thesis is humbly dedicated to them.

Contents

- 1 Introduction** **1**
- 1.1 Motivation and Overview 1
 - 1.1.1 Thesis Statement 2
 - 1.1.2 [Part I] Theory: Chapters 2, 3, 4, 5 3
 - 1.1.3 [Part II] Algorithms: Chapters 6, 7, 8, 9 5
 - 1.1.4 [Part III] Models: Chapters 10, 11 6
- 1.2 Contributions and Impact 7

- I Theory** **10**

- 2 Epidemic Thresholds: Static Graphs and Arbitrary Models** **12**
- 2.1 Introduction 12
- 2.2 Related Work 14
 - 2.2.1 Epidemic Thresholds 14
 - 2.2.2 Information Diffusion 15
 - 2.2.3 Cyber-physical infrastructures 15
- 2.3 Problem Formulation 16
- 2.4 Results 16
- 2.5 Proof Overview 18
 - 2.5.1 Our Terminology 19
 - 2.5.2 Our General Model 19
 - 2.5.3 Proof Sketch 21
- 2.6 Experiments 23
- 2.7 Implications 26
 - 2.7.1 Vulnerability of Networks—focus on eigenvalues 26
 - 2.7.2 Counter-intuitive Results 27
- 2.8 Impact 27
 - 2.8.1 Effective Immunization 28
 - 2.8.2 Evaluating ‘What-if’ Scenarios 28
 - 2.8.3 Accelerating Simulations 28
 - 2.8.4 Applications to Computer Networking 29

2.9	Conclusion	29
2.A	Notation	30
2.B	System Equations	30
2.C	Fixed point	32
2.D	The Jacobian	33
2.E	Eigenvalues of the Jacobian	34
	2.E.1 Eigenvalues of \mathbf{B}_1	35
	2.E.2 Eigenvalues of \mathbf{B}_3	36
2.F	Stability	37
	2.F.1 Case C1	37
	2.F.2 Case C2	37
3	Epidemic Thresholds: Time-varying Graphs	40
3.1	Introduction	40
3.2	Related Work	41
3.3	Problem Definitions	41
3.4	Epidemic Threshold on Time-varying Graphs	43
	3.4.1 The NLDS	43
	3.4.2 The Threshold	44
3.5	Salient Points	46
3.6	Experiments	46
3.7	Discussion—Generality of our results	48
3.8	Conclusion	48
4	Competing Viruses: Winner Takes All	49
4.1	Introduction	49
4.2	Related Work	51
4.3	Problem Formulation	51
	4.3.1 The propagation model	51
	4.3.2 Problem Statement	53
4.4	WTA: Results and Proofs	53
	4.4.1 Proof roadmap	54
	4.4.2 Special case: Clique Topology	56
	4.4.3 Special Case: Barbell Graph	57
	4.4.4 General Arbitrary Graph	58
4.5	Experiments	62
	4.5.1 Setup	63
	4.5.2 Simulation Results	64
	4.5.3 Case-Studies using Real Data	66
4.6	Discussion	67
4.7	Conclusions	68

5	Competing Viruses: Co-existence	69
5.1	Introduction	69
5.2	Related Work	70
5.3	Problem Formulation	71
5.3.1	The propagation model	71
5.3.2	Problem Statement	73
5.3.3	Model Formulation for a Clique	73
5.4	Results and Proofs	73
5.4.1	Formulating the problem	73
5.4.2	Results	74
5.5	Experiments	78
5.5.1	Setup	79
5.5.2	Simulation Results	79
5.5.3	Case-Studies using Real Data	80
5.6	Discussion	81
5.6.1	A general upper bound	81
5.6.2	Case-Study: Qualitative Analysis	82
5.6.3	Subtle Points	83
5.7	Conclusions	84
II	Algorithms	86
6	Complete Node-Removal	88
6.1	Introduction	88
6.2	Related Work	90
6.3	Problem Definitions (Static Graphs)	91
6.4	Background: Our Solution for Problem 1	92
6.4.1	‘ <i>Vulnerability</i> ’ Score	92
6.4.2	Justifications	93
6.5	Our Solution for Problem 2	93
6.5.1	Proposed ‘ <i>Shield-value</i> ’ Score	94
6.5.2	Justifications	94
6.6	Our Solution for Problem 3	96
6.6.1	Preliminaries	96
6.6.2	Proposed NETSHIELD Algorithm	96
6.6.3	Analysis of NETSHIELD	97
6.7	Experimental Evaluations (Static Graphs)	99
6.7.1	Data sets	99
6.7.2	Effectiveness	100
6.7.3	Efficiency	103
6.8	Immunization under time-varying graphs	105

6.8.1	Quality Metric	106
6.8.2	Proposed immunization policies	106
6.8.3	Experimental Setup	107
6.8.4	Results	108
6.8.5	Discussion	109
6.9	Conclusion	110
7	Fractional Immunization	111
7.1	Introduction	111
7.2	Related Work	114
7.3	Problem Formulation and Hardness result	115
7.3.1	Our proposed problem—MIN-CONN	116
7.3.2	MIN-CONN is NP-complete	118
7.4	Proposed Method—Overview	119
7.4.1	Algorithm EXHAUSTIVE	119
7.4.2	Algorithm SMART-ALLOC	120
7.5	Proposed Method—Theorems and proofs	120
7.5.1	Best single allocation—Details	120
7.5.2	Batched allocation—Details	121
7.6	Experiments	124
7.6.1	Setup	124
7.6.2	Effectiveness for MIN-CONN problem	126
7.6.3	Effectiveness for MAX-HEALTH problem	126
7.6.4	Scalability	128
7.6.5	Generality	128
7.7	Conclusion	129
8	General Edge Placement	130
8.1	Introduction	130
8.2	Problem Definitions	131
8.3	Proposed Algorithm for NETMELT	134
8.3.1	Edge Deletion vs. Node Deletion	134
8.3.2	Proposed K-EDGEDELETION Algorithm	136
8.3.3	Proofs and Analysis	137
8.4	Proposed Algorithm for NETGEL	138
8.4.1	Proposed K-EDGEADDITION Algorithm	138
8.4.2	Proofs and Analysis	139
8.5	Experimental Evaluations	140
8.5.1	Experimental Setup	141
8.5.2	Effectiveness of K-EDGEDELETION	141
8.5.3	Effectiveness of K-EDGEADDITION	144
8.5.4	Scalability	146

8.6	Related Work	147
8.7	Conclusion	147
9	Finding Culprits	150
9.1	Introduction	150
9.2	Preliminaries	152
9.2.1	Notation	153
9.2.2	The Susceptible-Infected Model	153
9.2.3	Minimum Description Length Principle	154
9.3	Our Problem Formulation	154
9.3.1	Cost of the Model	154
9.3.2	Cost of the Data given the Model	155
9.3.3	The Problem	157
9.4	Proposed Method	157
9.4.1	Best seed-set given number of seeds — ‘Exoneration’	157
9.4.2	Finding best single seed—Our Main Idea	158
9.4.3	Finding the best single seed—Justification	158
9.4.4	Finding best k-seed set	162
9.4.5	Finding a good ripple	162
9.5	Experiments	164
9.5.1	Experimental Setup	164
9.5.2	Effectiveness of NETSLEUTH in identifying How Many	166
9.5.3	Effectiveness of NETSLEUTH in identifying Which Ones	166
9.5.4	Scalability	167
9.6	Related Work	168
9.7	Conclusions	168
III	Models	169
10	Rise and Fall in Information Diffusion	171
10.1	Introduction	171
10.2	Background	174
10.3	Proposed Method	175
10.3.1	Base model - SPIKEM-BASE	176
10.3.2	With periodicity - SPIKEM	178
10.3.3	Additional details	179
10.4	Experiments	179
10.4.1	Q1: Explaining K-SC clusters	181
10.4.2	Q2: Matching <i>MemeTracker</i> patterns	182
10.4.3	Q3: Matching other data	184
10.4.4	Q4: Tail-part forecasts	185

10.5	Discussion - SPIKEM at work	185
10.5.1	“What-if” forecasting	185
10.5.2	Outlier detection	187
10.5.3	Reverse engineering	187
10.6	Related Work	188
10.7	Conclusions	189
11	Patterns amongst Competing Tasks	190
11.1	Introduction	190
11.2	Competing Tasks Model (CTM)	192
11.2.1	Justification	193
11.3	Experiments	194
11.3.1	CTM at Work	194
11.4	Related Work	196
11.5	Conclusions	196
IV	Conclusion	198
12	Conclusions and Future Directions	199
12.1	Summary of contributions	199
12.2	Vision and Future directions	201
12.2.1	Long Term Challenges	201

Chapter 1

Introduction

This thesis involves the study of propagation processes on large graphs. Will a specific YouTube video go viral? Given a who-contacts-whom network and a virus propagation model, can we predict whether there will be an epidemic? Which are the best nodes (people, computers etc.) to immunize, to slow down and prevent an epidemic as soon as possible? Such problems are central in surprisingly diverse areas: from *cyber security*, *epidemiology* and *public health*, product *marketing* to *information* dissemination. Answering these questions involves the study of aggregated dynamics over complex connectivity patterns. The proliferation of Internet and Web 2.0 and social networks like Facebook, Twitter, Flickr etc. coupled with more biological data and simulations has afforded the opportunity to study dynamical processes on a scale unimaginable before. Understanding such processes will eventually enable us to manipulate them for our benefit e.g., a better understanding of the dynamics of epidemic spreading over graphs allows us to devise more robust policies for immunization. Social-network websites like Facebook count more than 900 Million users and 1 Billion US Dollars in revenue. Hospital-acquired infections take more than 99 thousand lives and cost more than 5 Billion US Dollars per year. The societal impact of networked-collaboration during political events like ‘Arab Spring’ have also been well-documented. Hence research in this area, helping us answer questions like how information spreads through social media, and how to distribute a given amount of resource like antibiotics across hospitals, holds great scientific, social as well as commercial value.

This thesis gives new *theory*, better *algorithms* and improved *models* for propagation processes on large real-world networks. The next section gives an overview of the thesis, after which we present a summary of the major contributions.

1.1 Motivation and Overview

Graphs—also known as networks—are powerful tools for modeling processes and situations of interest in real-life like social-systems, cyber-security, epidemiology, biology etc. They are ubiquitous, from online social networks, gene-regulatory networks, to

router graphs. They effectively model a wide range of phenomena, as they expose local-dependencies and capture large-scale structure at the same time. For example, online social networks have become essential for marketing, collaborative action; gene-regulatory networks help in understanding the inner workings of the cell; modeling traffic on AS-router graphs helps us design a more efficient Internet. In addition, dynamical processes¹ over them can give rise to astonishing macroscopic behavior, leading to challenging and exciting research problems. How do contagions spread in population networks? Which group should we market to, for maximizing product penetration? How stable is a predator-prey ecosystem, given intricate food webs? How do rumors spread on Twitter/Facebook? Questions such as how blackouts can spread on a nationwide scale, or how social systems evolve on the basis of individual interactions, are all also related to dynamical phenomena on networks. ‘Big-Data’ is a natural and necessary part of research in this sphere. Although the actions of a particular individual or component may be too difficult to model, data mining, and machine learning can be applied to large groups or ensembles, which can yield effective models with the ability to predict future events. For example, modeling the behavior of every individual to a marketing strategy might be too difficult, but modeling the behavior of large and groups of people based on demographics and geography is feasible. Hence, we can try to answer even more complex issues using these models e.g., How should we distribute resources to control an epidemic? And these policies have to be designed in a way that they can be implemented on an extremely large-scale.

This thesis tackle several natural and fundamental problems in epidemic-style propagation (like, say ‘word-of-mouth’ viral marketing) on large real graphs. In addition, due to the sheer reach of the problems, an *inter-disciplinary* approach is vital here—this thesis involves work with collaborators spanning social media, medicine and public health, mobile and internet networking and online services and operations. Our research has been inherently multi-pronged combining:

- I (Theory) Analyzing theoretical models of propagation processes
- II (Algorithms) Developing scalable algorithms to manage the processes
- III (Models) Using massive datasets to make better models

Thus this thesis essentially breaks down into three parts, each of which we summarize in the next few subsections. We want to highlight that these parts are all symbiotic and closely related. For example, once we collect and learn models of a disease from real-data, we can predict its tipping point (through analysis) and subsequently leverage it for immunization (increase the tipping point as much as possible).

1.1.1 Thesis Statement

Substantially better algorithms can be designed for cascade management and immunization through careful and novel analysis of virus propagation models.

¹Intuitively, where the state (or action) of an agent depends on the states (actions) of its neighbors.

More specifically, we systematically analyze fundamental epidemic models in a variety of situations (static/dynamic graphs, single/multiple viruses) to better understand the effect of graph topology on various propagation processes (like epidemic thresholds) and subsequently leverage our results to carefully design better algorithms for many tasks like immunization (like SMART-ALLOC). Finally, using our insights, we build better models for describing propagation scenarios (like SPIKEM) to match real data.

1.1.2 [Part I] Theory: Chapters 2, 3, 4, 5

This part of the thesis is devoted to gaining a deeper understanding of abstract epidemic models. Models help us abstract out the process and allow us to reason more generally about them. In this part, we tackled important questions like understanding *the tipping point* behavior of epidemics, predicting *who-wins* among competing viruses/products, which have immediate and broad applications, like selecting targets for advertising and marketing and selecting people to inoculate to stop an epidemic. In contrast to previous work, our analysis focuses on *arbitrary* underlying graphs, leading to more readily applicable results.

Chapter 2 (Epidemic thresholds for static graphs): The main question we answer is: will there be an epidemic, given the graph and the virus propagation model? We show (see Theorem 2.1) that the threshold condition is (λ_1 is the first eigenvalue of the connectivity matrix, C is a virus-model dependent constant):

$$\lambda_1 \cdot C < 1,$$

for (a) *any* graph; and (b) *all* propagation models in standard literature (more than 25 from canonical texts, including the AIDS virus *H.I.V.*). Our result de-couples the effect of the topology and the virus model, and also unifies and subsumes older results, which mostly focused on special graphs or virus models. We are the **first** to show the epidemic threshold on *arbitrary* graphs and almost *any* virus propagation model. Our discovery has broad implications and applications like faster epidemiological simulations and blog cascades (like the award-winning Independent Cascade model is a special case of our generalization).

Chapter 3 (Epidemic thresholds for dynamic graphs): Social-contacts are not constant, and therefore it is more realistic to have graphs which change with time (say, day vs. night connectivity). While static graphs have been studied for a long time, with numerous analytical results, time-evolving networks are so hard to analyze, that most existing works are simulation studies. Most existing works are simulation studies, as propagation models on time-evolving networks are so hard to analyze. We show that the epidemic threshold of the “flu-like” SIS model on *any set* of time-varying, arbitrary graphs depends only on the largest eigenvalue of a so-called ‘system’ matrix (see Theorem 3.1).

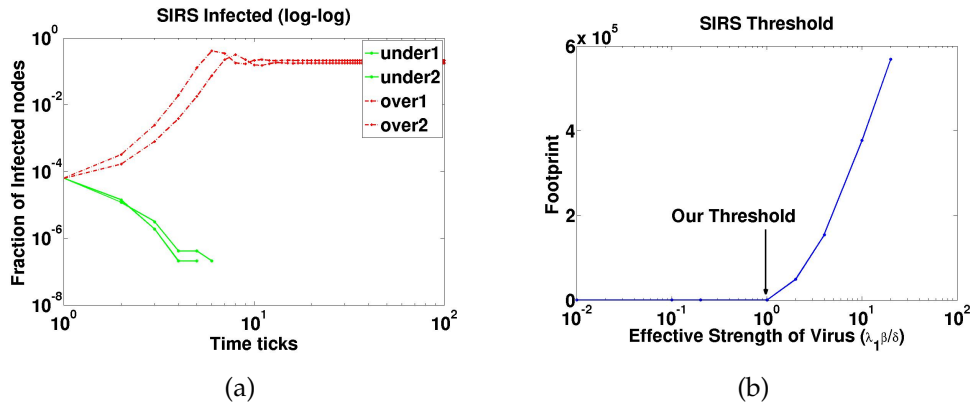


Figure 1.1: The tipping-point exactly matches our prediction: simulation results on a massive social-contact graph PORTLAND (31 mil. edges, 1.5 mil. nodes) and the SIRS model (temporary immunity like pertussis). (a) Plot of Infected Fraction of Population vs Time (log-log). Note the qualitative difference in behavior- *under* (green) the threshold and *above* (red) the threshold. (b) *Footprint* (expected final epidemic size) vs Effective Strength (lin-log). Notice our prediction is exactly at the take-off point.

Chapter 4 (Mutually exclusive competing viruses): In this chapter, we shift our focus to competing viruses spreading over networks. Given two competing products such as iPhone/Android, and ‘word of mouth’ adoption of them, what will happen in the end? Will they split the market, in proportion to their quality? i.e. which product will ‘win’, in terms of highest market share? This question is of interest in numerous other settings too, e.g., the common flu versus avian flu, competing memes, theories and so on. One may naïvely expect that the better product (stronger virus) will just have a larger footprint, proportional to the quality ratio of the products (or strength ratio of the viruses). We prove the surprising result (see Theorem 4.1) that, under realistic conditions, for *any* graph, the stronger virus completely wipes-out the weaker virus (‘winner-takes-all’). We demonstrate it through case-studies using real data too.

Chapter 5 (Co-existence with competing viruses): Following from the previous chapter, a natural question is to understand what happens when the competing viruses are not mutually exclusive? For example, using one web-browser (say I.E.) does not automatically imply not using the other (say Chrome). We show (see Theorem 5.1) that there is a phase-transition: if the competition is harsher than a critical threshold, then ‘winner-takes-all’, otherwise, the weaker virus survives. Our contributions [BPRF12] include the problem definition (which is unique even in epidemiological literature), and experiments on real-data demonstrating our result.

1.1.3 [Part II] Algorithms: Chapters 6, 7, 8, 9

This part of the thesis is devoted to developing *fast and effective algorithms* for a variety of tasks w.r.t. propagation: immunization, edge-placement and finding culprits of epidemics. Such problems naturally arise in epidemiology ('vaccination programs'), social media ('detecting rumor sources') and cyber security ('designing worms'). Interestingly, our previous work on thresholds in various settings above give a clear guideline for controlling (harmful viruses) or speeding-up (product marketing) propagation via network manipulation: minimize (or maximize) the leading eigenvalue of a suitable matrix (adjacency matrix in case of static graphs; the so-called system matrix in case of dynamic graphs). This is in contrast to previous work, where complex optimization functions were used. Unfortunately, we also prove that our problems are computationally hard (NP-complete). We exploit the task-specific structure to get fast (*linear-time* in edges and the budget) and accurate algorithms, substantially improving the state-of-the-art.

Chapter 6 (Immunization as node-removal): Given a large network, like a computer communication network, which k nodes should we remove (or monitor, or immunize), to make it as robust as possible against a computer virus attack? Making careful approximations, we exploit the *submodular* structure of the set of possible solutions, getting a provably near-optimal algorithm NETSHIELD (see Algorithm 1), which outperforms many methods by more than **7 orders** of magnitude in running time, and competitors like the well-known acquaintance immunization in quality of solutions. A similar question arises in case of viral propagation over time-varying dynamic graphs. We develop fast heuristics for complete immunization in case of time-varying graphs as well and demonstrated their effectiveness (see Section 6.8).

Chapter 7 (Fractional Immunization): Given a fixed amount of medicines with partial impact, how should they be distributed? Collaborating with domain experts at UMichigan, we studied controlling the spread of bacteria between hospitals through patient transfers, by distributing scarce infection-control resources (which only have partial impact on any hospital). it is NP-complete, and develop SMART-ALLOC, a *near-optimal* and *fast* algorithm. SMART-ALLOC runs in *seconds* on commodity hardware, as opposed to *weeks* required for other approaches. Most importantly, when applied on real hospital patient-transfer networks like US-MEDICARE, it results in *6 times* fewer infections. Figure 1.2 illustrates our results (each resource roughly halved the susceptibility of a hospital and the same amount (200) were distributed).

Chapter 8 (General Edge-placement): Which people should be introduced to each other, to maximize the spread of a crucial piece of information? Which people should be 'un-friended' to contain dissemination of malware over Facebook? We study the edge-placement problem: which edges should we add or delete in order to speed-up or contain a dissemination? For addition of edges, things are even more challenging

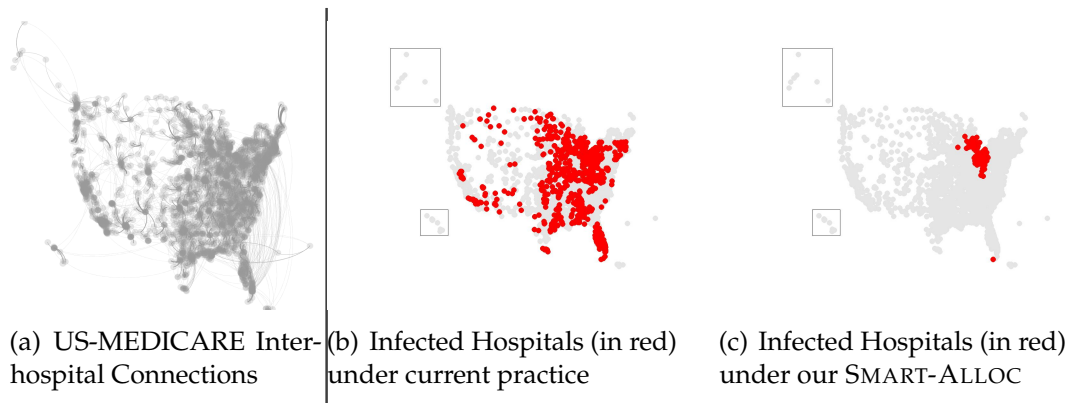


Figure 1.2: Our proposed SMART-ALLOC method has $6x$ fewer infections (red circles): (a) The US-MEDICARE network of hospitals overlaid on a map (b) Infected hospitals after a year (365 days) under current practice. (c) Similarly, under SMART-ALLOC. The current practice allocates equal amounts of resource to each hospital.

because of its intrinsic quadratic time complexity. We propose *effective* and *near linear-time* algorithms to solve these problems. We also study the two problems and our methods theoretically, the accuracy and complexity of our methods, and the equivalence between different strategies (edge vs. node-deletion). To the best of our knowledge, we are the **first** to study the edge-placement problem.

Chapter 9 (Finding culprits of epidemics): Can we identify sources of rumors on Twitter? Or given a snapshot of a large graph, in which an infection has been spreading for some time, can we reliably identify those nodes (both in number and identity) from which the infection started to spread? In this chapter we answer this question affirmatively, and give an efficient method called NETSLEUTH for the well-known Susceptible-Infected virus propagation model, which *automatically* finds out both the number and identity of the seeds which best-describe the epidemic. Experimentation on our method [PVF12] shows high accuracy in the detection of seed nodes, in addition to the correct automatic identification of their number. Moreover, we show NETSLEUTH scales *linearly* in the number of nodes of the graph, in contrast to existing methods.

1.1.4 [Part III] Models: Chapters 10, 11

In this part, we study numerous real-datasets to build better models in domains such as propagation of memes in online media and competing tasks in everyday life. We also show, as a bonus, how to use such models for varied challenging applications like forecasting trends activity and spotting outliers like telemarketers.

Chapter 10 (Rise and fall patterns): While models in epidemiology have been widely studied and accepted, the models describing exactly how information diffuses in online

media is uncertain. Here we ask a very simple question: How quickly does a piece of news spread over these media? How does its popularity diminish over time? Does the rising and falling pattern follow a simple universal law? We propose SPIKEM [MSP⁺12], a concise yet flexible model, which *generalizes* and *unifies* previous models and observations, and excels at challenging tasks like *forecasting*, *spotting anomalies* etc. We show the power of SPIKEM through the analysis of more than 7.2GB of real data.

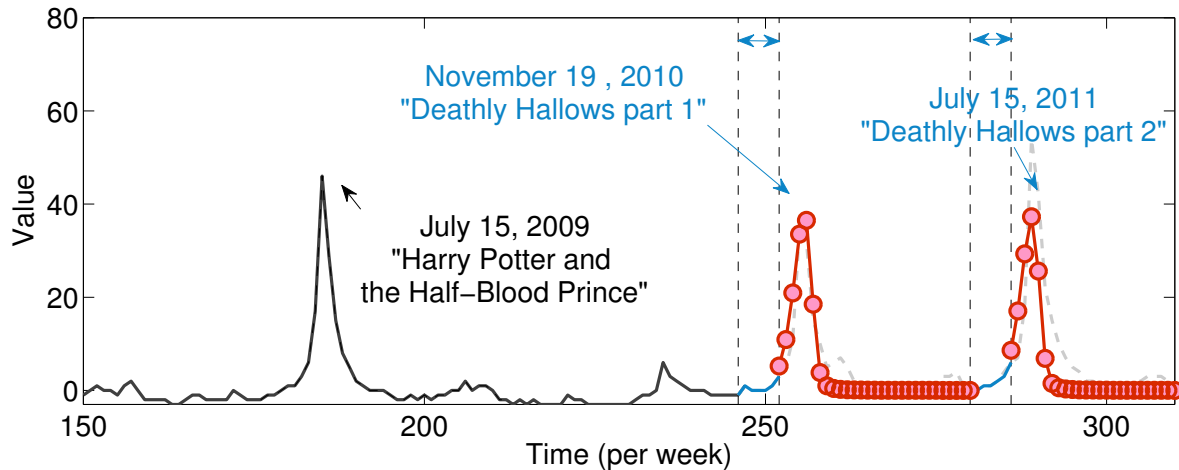


Figure 1.3: SPIKEM at work: Results of “what-if” forecasting for the Harry Potter series. We trained parameters by using (a) the first spike around July 15, 2009 (black solid line), and (b) access volume two months before the release (blue lines with double arrows around time $n = 250, 280$) and then, forecasted the following two spikes (red lines).

Chapter 11 (Competing Tasks): If Alice has double the friends of Bob, will she also have double the phone-calls (or wall-postings, or tweets)? Analyzing data containing *millions of users*, we show that the answer to the question is a *power-law*: sub-linear, or super-linear, for a wide variety of diverse settings: tasks in a phone-call network, like count of friends, count of phone-calls, total count of minutes; tasks in a twitter-like network, like count of tweets, count of followees etc. Additionally, based on competing species theories, we give a simple “competing tasks” model (CTM), that leads exactly to power-law relationships between task-frequencies, and show how to use it to spot *telemarketers*.

1.2 Contributions and Impact

We give the major contributions and impact of the work in this thesis next. We are arguably the *first* to present a systematic study of propagation and immunization of *single* as well as *multiple* viruses on *arbitrary, real* and *time-varying* networks as the vast

majority of the literature focuses on structured topologies, cliques, and related un-realistic models.

Theory

- **Eigenvalues for Epidemic Threshold:** We are the **first** to show the epidemic threshold on *arbitrary* graphs and almost *any* virus propagation model. Our eigenvalue result generalizes and unifies previous results and has broad implications and applications like faster epidemiological simulations. We also derive the **first** closed formula for *any* set of arbitrary time-varying graphs. In contrast, most past work has used simulations.
- **Winner-Takes-All for Competing Viruses:** We are the **first** to prove for *arbitrary* graphs that winner-takes-all in competing viruses/products. Additionally, we extend this problem to mutually-interacting viruses to show a phase-transition.
- ★ Impact: Our paper on epidemic thresholds on static graphs [PCF⁺11] was selected for one of the best papers of the conference. Our results have been used to enable other important tasks like anomaly detection and graph modeling [AMF10], and immunization (see Part II of this thesis). They are also being incorporated into *FRED*, an epidemiological simulator developed by MIDAS.

Algorithms

- **Dramatically better Immunization:** Our algorithms such as NETSHIELD and SMART-ALLOC solve the complete and fractional immunization problems respectively, achieving significant savings. NETSHIELD outperformed many methods by more than **7 orders** of magnitude in running time, and competitors like the well-known acquaintance immunization in quality of solutions. Additionally, on real hospital patient-transfer networks like US-MEDICARE, SMART-ALLOC achieves up to **6x fewer** infections and **30,000x speed-up**, over current practice and ad-hoc heuristics. In contrast, the current practice in control of highly resistant organisms via patient transfers has been largely focused within individual hospitals.
- **Parameter-free Culprits detection:** Our algorithm NETSLEUTH is the **first linear-time** algorithm (in edges and nodes) for both identifying the set of culprits for which best describes a given snapshot of the epidemic and for **automatically** selecting the best number of seed nodes—in contrast to the state of the art (which are at least quadratic, and require the number of seeds as input).
- ★ Impact: Our results and algorithms have been incorporated into undergraduate courses (UPitt Summer Program) and slides sought-after for graduate courses (Xifeng Yan, UCSB) in universities, and have appeared in ACM Crossroads.

Models

- **Unifying models for online diffusion:** We develop SPIKEM, a powerful model

to explain the rise and fall patterns of information diffusion which **unifies** and includes earlier patterns and models, is succinct, matches behavior of numerous real datasets and can be used to **forecast**, answer '**what-if**' scenarios and even *reverse-engineer* epidemics.

- **Explaining power-laws in competing tasks:** We develop CTM, an intuitive model to explain the prevalence of **super-linear** relationships between the frequencies of various competing tasks observed in real-datasets, and use it to spot outliers like **telemarketers**.

Part I
Theory

Overview

This part deals with the *analysis* of epidemic-like models on networks. We answer two main questions here:

- **Epidemic Thresholds:** Given a network of who-contacts-whom or who-links-to-whom, will a contagious virus (or product or meme) spread and ‘take-over’ (cause an epidemic) or die-out quickly? What will change if nodes have partial, temporary or permanent immunity? The epidemic threshold is the minimum level of virulence to prevent a viral contagion from dying out quickly and determining it is a fundamental question in epidemiology and related areas. For the static graphs, we show that the threshold condition is (λ_1 is the first eigenvalue of the connectivity matrix, C is a virus-model dependent constant): $\lambda_1 \cdot C < 1$, for (a) *any* graph; and (b) *all* propagation models in standard literature (more than 25 from canonical texts, including the AIDS virus *H.I.V.*). Additionally, for any set of *arbitrary time-varying* graphs, we show that the threshold depends only on the largest eigenvalue of a so-called ‘system’ matrix.
- **Competing Viruses:** Given two competing products such as iPhone/Android, and ‘word of mouth’ adoption of them, what will happen in the end? Will they split the market, in proportion to their quality? We prove the surprising result that, under realistic conditions, for *any* graph, the stronger virus completely wipes-out the weaker virus (*winner-takes-all*). Further, we extend this analysis for viruses interacting more subtly and prove the existence of a *phase transition* for co-existence. We demonstrate these results through case-studies using real data too.

These are natural and fundamental questions, and our results in this part have broad applications, like faster epidemiological simulations, better immunization algorithms and prediction (some of which we will see also in the subsequent parts of this thesis).

Chapter 2

Epidemic Thresholds: Static Graphs and Arbitrary Models

Given a network of who-contacts-whom or who-links-to-whom, will a contagious virus (or product or meme) spread and ‘take-over’ (cause an epidemic) or die-out quickly? What will change if nodes have partial, temporary or permanent immunity? The epidemic threshold is the minimum level of virulence to prevent a viral contagion from dying out quickly and determining it is a fundamental question in epidemiology and related areas. Networks with lower thresholds are susceptible to weak contagions. Conversely, raising the threshold (e.g., through immunization), protects the network against attacks. Most earlier work focuses either on special types of graphs or on specific epidemiological/cascade models. In this chapter, we are the first to show the G_2 -threshold (twice generalized) theorem, which nicely de-couples the effect of the topology and the virus model. Our result unifies and includes as special case older results and shows that the threshold depends on the first eigenvalue of the connectivity matrix, (a) for any graph and (b) for all propagation models in standard literature (more than 25, including *H.I.V.*).

Our discovery has broad implications for the vulnerability of real, complex networks, and numerous applications, including viral marketing, blog dynamics, influence propagation, easy answers to ‘what-if’ questions, and simplified design and evaluation of immunization policies. We also demonstrate our result using extensive simulations on real networks, including on one of the biggest available social-contact graphs containing more than *31 million* interactions among more than *1 million* people representing the city of Portland, Oregon, USA.

2.1 Introduction

Given a social or computer network, where the links represent who has the potential to infect whom, what can we say about its epidemic threshold? That is, can we determine

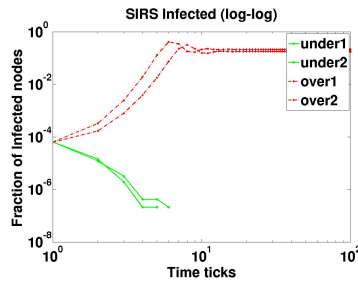


Figure 2.1: Qualitatively different infection time-series curves (Fraction of Infected population vs Time) for the SIRS model (temporary immunity, like pertussis) on a large contact-network. What is the condition that separates the two regimes - red (epidemic) vs green (extinction)?

whether a small infection can ‘take-off’ and create an epidemic? What will change if the nodes have permanent, temporary or no immunity? Both the underlying contact-network (or the population structure) and the particular cascade (propagation) model should intuitively play an important role in the spread of contagions (viruses/memes/products). Finding the epidemic threshold for an arbitrary network is an important and fundamental question in epidemiology and related areas. For instance, Figure 2.1 shows the simulation output after running the SIRS model (Susceptible-Infectious-Recovered-Susceptible which models diseases with temporary immunity like pertussis) on a large contact-network for different values of the virulence of the virus (achieved by tuning the parameters of the model). We can clearly see two different regimes - the fast die-out green regime and the steady-state epidemic red regime. This chapter deals with finding the condition which separates these two regimes in SIRS, as well as in *all* other virus propagation models in standard literature [Het00, EK10], on *arbitrary* contact-networks.

Much of previous work focuses on either special types of graphs (typically cliques [KW93], block-structure and hierarchical graphs [HY84] and random power-law graphs [PSV01]) or on specific epidemiological models [CWW⁺08]. We unify and include as special-case older results in two orthogonal directions and show:

- *De-coupling*: the threshold condition *separates* the effect of topology and the virus model,
- *Arbitrary Topology*: the threshold depends on the first eigenvalue of the connectivity matrix,
- *Arbitrary VPM*: the threshold depends on one constant that completely characterizes the virus propagation model (VPM)

Our result has numerous applications and immediate implications (see § 2.8) including easy answers to ‘what-if’ questions and simplified design and evaluation of immunization policies. Moreover, a variety of dynamic processes on graphs are modeled like epidemic spreading and hence our result applies to many of them. For example, the linear-cascade model [KKT03] is essentially the SIR model (Susceptible-Infected-Recovered, models chicken pox, see Figure 2.2 (left inset) for state diagram); also, so-called threshold models (like Granovetter’s model [Gra78]) in sociology are similar in reality to cascade

models [DW04]. In contrast to harmful viruses, the propagation of some contagions may in fact be *desirable* e.g., dissemination of a product or an idea in a network of individuals. For example, the Bass model [Bas69] fits product adoption data using parameters for pricing and marketing effects. However it ignores topology; it simply assumes that all adopters have equal probability of influencing non-adopters. Instead, using our result, a more refined picture can be constructed of when a product gains massive adoption on a social network (equivalent to an “epidemic”).

Several VPMs have direct applications in modeling computer and email viruses [Kle07, HMM03]. In these cases, more so than the biological ones, it is easier to get the entire underlying network. Hence our threshold results can be used to make the network more robust by “immunizing” a few carefully chosen computers in the network (like installing a firewall on them). Another application is the efficient spreading of software patches over a computer network. The patches behave like computer worms [VGKG08] and can help defend against other malicious worms. Given full knowledge of the router-network involved, we can then estimate how “infectious” the patch-worm has to be (say by increasing the number of probes for possible hosts before dying out) to at least initiate an “epidemic” w.r.t. the patch. Additionally, we can help determine the vulnerability and consequently the cost of not patching parts of the network. Various epidemic models have also been used to model blog cascades which can now be applied to arbitrary graphs e.g., to study the propagation of memes through blogs [LBK09].

The rest of the chapter is organized as follows: we first give the related work in § 2.2, then formulate the problem (§ 2.3) and state our main result (§ 2.4), give a proof roadmap and example (§ 2.5) and then show simulation experiments (§ 2.6) to demonstrate the result. We discuss the broad implications and many applications of the result in § 2.7 and § 2.8. We then conclude (§ 2.9) and finally give a detailed proof in the Appendix.

2.2 Related Work

We review related work here, which can be categorized into three parts: epidemic thresholds, information diffusion and cyber-physical infrastructures. None of these works generalize in two directions: for arbitrary propagation models and arbitrary networks.

2.2.1 Epidemic Thresholds

Canonical texts for epidemiology include [AM91, Het00]. The most widely-studied epidemiological models include the so-called *homogeneous models* (for example, the SIR model was introduced by McKendrick in the 1920’s [McK25]), which assume that every individual has equal contact to others in the population and that the rate of infection is determined by the density of the infected population. Kephart and White [KW93] were among the first to propose epidemiology-based models (the KW model) to analyze

the propagation of computer viruses on homogeneous networks. However, there is overwhelming evidence that real networks including social networks, router and AS networks [FFF99] etc. follow a power law structure instead. Pastor-Satorras and Vespignani [PSV01] studied viral propagation for random power-law networks, and showed low or non-existent epidemic thresholds, meaning that even an agent with extremely low infectivity could propagate and persist in the network. They use the “mean-field” approach, where all graphs with a given degree distribution are considered equal. There is no particular reason why all such graphs should behave similarly in terms of viral propagation. In a recent work, Castellano and Pastor-Satorras [CPS10] empirically argue that some special family of random power-law graphs have a non-vanishing threshold under the SIR model in the limit of infinite size, but provide no theoretical justification.

Newman [New05a, New02] mapped the SIR model to a percolation problem on a network and studied thresholds for multiple competing viruses on special random graphs. Finally, Chakrabarti et.al. [CWW⁺08] and Ganesh et.al [GMT05] gave the threshold for the SIS model on arbitrary undirected networks. Hence, *none* of the earlier work focuses on epidemic thresholds for *arbitrary* virus propagation models on *arbitrary*, real graphs.

2.2.2 Information Diffusion

There is a lot of research interest in studying dynamic processes on large graphs, (a) blogs and propagations [GGLNT04, KNRT03, KKT03, RD02], (b) information cascades [BHW92, GLM01, Gra78, GRLK10, ZWF⁺11] and (c) marketing and product penetration [Rog03, LAH06]. Competitive cascades have been studied in [PBS10, PBRF12]. Various optimization problems have also been studied on such processes like influence maximization [KKT03, CWW10, SKOM12] and finding effectors [LTGM10]. These dynamic processes are all closely related to virus propagation, with many directly based on epidemiological models [Bas69, KKT03] e.g., the award-winning linear-cascade model [KKT03] is a *special* case of our model : specifically it is essentially a SIR model with $\delta = 1$ and all our results carry through.

2.2.3 Cyber-physical infrastructures

Cascade models have also been applied to real-world networks to understand network robustness in cyber-physical infrastructures, i.e., the ability of a network to continue it’s function in light of failures. The exact nature of a network’s “function” varies from network-to-network and is typically determined during their design phase. Models related to the spread of disease—similar to those modeled herein—have been used to analyze the robustness of networks against node failures, for instance, see Chakrabarti et al. [CLF⁺07]. Another such work is Buldyrev et al. [BPP⁺10], which explores cascading failures in coupled power and data networks. Their model is based on a percolation model common in statistical physics, and can be shown equivalent to the SIR model we describe later in the chapter.

Table 2.1: Common Terminology

Term	Definition
VPM	virus-propagation model
NLDS	non-linear discrete-time dynamical system
β	attack/transmission probability over a contact-link
δ	healing probability once infected
γ	immunization-loss probability once recovered (in SIRS) or vigilant (in SIV, SEIV)
ϵ	virus-maturation probability once exposed hence, $1 - \epsilon$ is the virus-incubation probability
θ	direct-immunization probability when susceptible
\mathbf{A}	adjacency matrix of the underlying undirected contact-network
N	number of nodes in the network
λ_1	largest (in magnitude) eigenvalue of \mathbf{A}
s	effective strength of an epidemic model on a graph with adjacency matrix \mathbf{A}

2.3 Problem Formulation

Table 2.1 and Table 2.2 list common terminology and describe some of the epidemic models we will be using in the work. We use the term ‘cascade model’ and ‘virus propagation model’ interchangeably. We next state formally the problem we address in this chapter:

Problem 2.1. Epidemic Threshold

Given: A undirected unweighted graph G , and a virus propagation model (VPM) and its parameters (e.g., β and δ for SIR).

Find: A condition under which will an infection will die out and not cause an epidemic on the graph.

2.4 Results

The epidemic threshold is usually defined as the minimum level of virulence to prevent a viral contagion from dying out quickly [AM91, Het00, BBV10, Kle07]. In order to standardize the discussion of threshold results, we express the threshold in terms of the normalized *effective strength*, s , of a virus which is a function of the *particular* propagation model and the *particular* underlying contact-network. So we are ‘above threshold’ when $s > 1$, ‘under threshold’ when $s < 1$ and the threshold or the tipping point is reached when $s = 1$. The effective strength s can be thought of as the basic reproduction number R_0 frequently used in epidemiology [Het00, AM91]. It (s) is then very roughly, the “net”

Table 2.2: Some Virus Propagation Models (VPMs)

Model	Description
SIS	'susceptible, infected, susceptible' VPM - no immunity, like flu
SIR	'susceptible, infected, recovered' VPM - life-time immunity, like mumps
SIRS	VPM with temporary immunity
SIV	'susceptible, infected, vigilant' VPM - immunization/vigilance with temporary immunity
SEIR	'susceptible, exposed, infected, recovered' VPM - life-time immunity <i>and</i> virus incubation
SEIV	VPM with vigilance/immunization with temporary immunity <i>and</i> virus incubation

generalized R_0 for the virus model and an arbitrary graph and is the quantity which determines the tipping point of an infection over a contact-network. Our main result is:

Theorem 2.1 (G2-threshold theorem). *For any virus propagation model (satisfying our general initial assumptions; see Section 2.5 for details) operating on an arbitrary undirected graph with adjacency matrix \mathbf{A} and largest eigenvalue λ_1 , the virus will get wiped out if:*

$$s < 1 \tag{2.1}$$

where, s (the effective strength) is:

$$s = \lambda_1 \cdot C_{\text{VPM}} \tag{2.2}$$

and C_{VPM} is an explicit constant dependent on the virus propagation model. Hence, the tipping point is reached when $s = 1$.

Proof. We give a roadmap in the next section and a detailed proof in the Appendix. \square

Firstly, note that our result separates out the effect of the network and the VPM. Secondly, our result subsumes older results on (a) contact-networks, and (b) VPMs as special cases. Results on contact-networks like cliques (everybody contacts everybody else: $\lambda_1 = N - 1$, N is the number of nodes in the graph), random Erdős-Rényi graphs with expected degree d ($\lambda_1 = d$), 'homogeneous' graphs [KW93], power-law/scale-free graphs [PSV01], structured hierarchical (near-block-diagonal) topologies [HY84] (people within a community contact all others in this community, with a few cross-community contacts) etc. are special cases. Likewise, all standard virus propagation models [Het00, EK10] are specific instantiations of the generalized model used in our theorem (see Figure 2.2; more later).

Table 2.3 lists a few of our threshold expressions after applying our result on some standard epidemic models. The popular models listed include SIS (no immunity, like flu,

Table 2.3: Threshold Results for Some Models.

Models	Effective Strength (s)	Threshold (tipping point)
SIS, SIR, SIRS, SEIR	$s = \lambda_1 \cdot \left(\frac{\beta}{\delta} \right)$	$s = 1$
SIV, SEIV	$s = \lambda_1 \cdot \left(\frac{\beta\gamma}{\delta(\gamma+\theta)} \right)$	
SI ₁ I ₂ V ₁ V ₂ (\sim H.I.V.)	$s = \lambda_1 \cdot \left(\frac{\beta_1\nu_2 + \beta_2\epsilon}{\nu_2(\epsilon + \nu_1)} \right)$	

SIS (*susceptible/infected/susceptible*) has no immunity (like flu), SIR (*susceptible/infected/recovered*) has permanent immunity (like mumps), SIRS has temporary immunity (like pertussis) while SEIR (*susceptible/exposed/infected/recovered*) has additional virus incubation and SI₁I₂V₁V₂ has been used to model some H.I.V. infections [2]. SEIV and SIV are two useful generalizations. β is the attack/transmission probability over a contact link, δ is the healing probability, γ is the immunization-loss probability, $(1 - \epsilon)$ is the virus incubation probability and θ is the direct-immunization probability when susceptible (see Figure 2). Our result is a general one and these models just highlight its ready applicability to standard VPMs in use.

Susceptible-Infected-Susceptible), SIR (permanent immunity, like mumps, Susceptible-Infected-Recovered), SIRS (temporary immunity, like pertussis), SEIR (virus incubation in addition to permanent immunity) etc. (note that models like SI inherently don't have an epidemic threshold as all nodes will eventually get infected on any graph - hence our work doesn't apply to them).

Table 2.3 also lists our SEIV model (Susceptible-Exposed-Infected-Vigilant) which itself generalizes almost all models from [Het00] (SIS with $\epsilon = 1, \gamma = 1, \theta = 0$; SIR with $\epsilon = 1, \gamma = 0, \theta = 0$; SIRS with $\epsilon = 1, \theta = 0$ and so on). Using our proof, we get that the effective strength for SEIV is $s = \lambda_1 \cdot \frac{\beta\gamma}{\delta(\theta+\gamma)}$ (as before the virus dies out if $s < 1$). Note that this implies that increasing β (the attack probability) strengthens the virus. At the same time, decreasing the healing probability δ also strengthens the virus. Finally, decreasing θ (the direct immunization probability) and increasing γ (the immunization loss probability) also makes the virus stronger. All of these fit with intuition - in fact, the usefulness of our result is partly in enabling us to see these complex effects on the virus strength very clearly. We discuss some subtler implications later in Section 2.7. We discuss our terminology, general model and proof sketch next.

2.5 Proof Overview

We first construct a generalized model ($S^*I^2V^*$ - arbitrary number of susceptible and vigilant states, two infectious states) that is powerful enough to generalize all the practical VPMs (and more) and satisfies our very general assumptions, while still being

mathematically tractable (Figure 2.2). We then approximate our general model using a discrete time non-linear dynamical system and transform the tipping point question into a stability problem of the dynamical system at an appropriate equilibrium point. We give the overview and roadmap here. As mentioned before, the full proof can be found in the Appendix.

2.5.1 Our Terminology

Note that any VPM has some states and the choice of which states to include in a model depends on the particular contagion characteristics. Yet, we can think of every model as having states essentially in any of the following fundamental broad classes:

1. *Susceptible Class*: Nodes in such a state can get infected by any neighboring node (in the contact-network) who is infectious.
2. *Infected Class*: In a state of this class, the node is infectious in the sense that it is capable of transmitting the infection to its neighbors. Note that each such state will have a *transmissibility* parameter (e.g., β in the SIR model for the infectious state I). Thus this can include models with transmissibility parameter = 0 i.e. they are ‘exposed’ but not infectious (e.g., the E state in the SEIR model is a state which is in the Infected class in the sense that it can potentially cause infections but is not by itself infectious).
3. *Vigilant/Vaccinated Class*: Nodes in any of the states in this class cannot get infected nor can they potentially cause infections. States like R in SIR (the recovered/died state where the node gets permanent immunity/dies and hence does not participate in the epidemic further), M in MSIR (the passive immune state), etc. are conceptually of the Vigilant type.

2.5.2 Our General Model

Using our terminology above, we can now describe the generalized model we used in Theorem 2.1: $S^*I^2V^*$ (arbitrary number of susceptible and vigilant states, two infectious states). As our general characterization, $S^*I^2V^*$ is powerful enough to seamlessly capture all the practical models (and more) like SIS, SIR, SIRS, SEIR, SERIS, MSIR, MSEIR etc. [Het00, EK10], including H.I.V. [AM91], while being tractable enough to yield simple threshold equations. Figure 2.2 shows the state diagram under $S^*I^2V^*$ for a node in the contact-network together with the assumptions on the transitions. The red-curved arrow indicates exogenous (*graph-based*) transition caused by infectious neighboring nodes while all other transitions are endogenous, caused by the node itself with some probability. We have shown only cross-class transitions and their types. We make two assumptions:

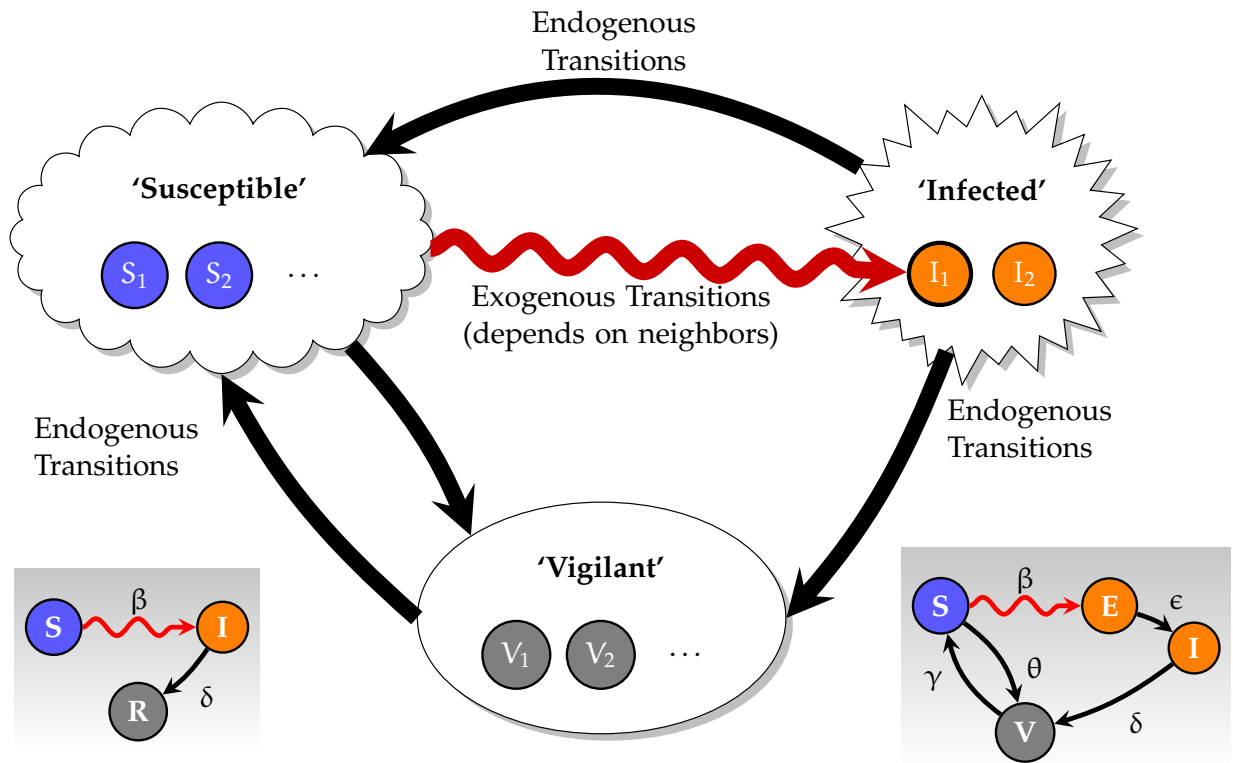


Figure 2.2: State Diagram for a node in the graph in our generalized model $S^*I^2V^*$ - it is *not* a simple Markov chain. There are three classes (types) of states - Susceptible (healthy but can get infected), Infected (capable of transmission) and Vigilant (healthy and can't get infected). Within-class transitions not shown for clarity. Red-curvy arrow indicates exogenous i.e. *graph-based* transition affected only by the neighbors of the node, all other transitions are *endogenous* (caused by the node itself with some probability at every time step). **(Left Inset)** *Special case:* Transition diagram for the SIR (Susceptible-Infected-Recovered) model. **(Right Inset)** Another special case: Transition diagram for the SEIV (E stands for exposed but not infectious) model. SEIV itself generalizes almost all models from [Het00] (SIS with $\epsilon = 1, \gamma = 1, \theta = 0$; SIR with $\epsilon = 1, \gamma = 0, \theta = 0$; SIRS with $\epsilon = 1, \theta = 0$ and so on).

1. *Infection through Neighbors*: The only way to get infected is through your neighbors i.e. there is no path to a state in the Infected class from a state in the Susceptible class composed solely of endogenous transitions.
2. *Starting Infected State*: For the few models that have more than one infectious state, any exogenous (graph-based) transition always results in a transition from a state in the Susceptible class to the I_1 state. Note that this assumption is trivially obeyed for a vast majority of models (with only one infected state).

Figure 2.2 (Left Inset) shows the popular SIR model as an instantiation of our general model $S^*I^2V^*$. Also, Figure 2.2 (Right Inset) shows an instantiation in the form of our SEIV model (Susceptible-Exposed-Infected-Vigilant). Figure 2.3 shows the generalization hierarchy for some common epidemic models and our main generalization $S^*I^2V^*$. The brown colored nodes denote standard VPMs found in literature while the blue colored nodes denote our generalizations. Each VPM is a generalization of all the models below it e.g., SIV is a generalization of SIRS, SIR and SIS.

2.5.3 Proof Sketch

We define the vector $\tilde{\mathbf{P}}_t$ such that it specifies the state of the system at time t ; the exact definition will differ from model to model but it effectively encodes the probability of each node in the graph of being in any given state at time t . Suppose the virus-propagation model has m (s_1, s_2, \dots, s_m) states (e.g., $m = 3$ for the SIR model with states $s_1 = S, s_2 = I$ and $s_3 = R$) and it operates on a graph of N nodes. Consider then a column vector $\tilde{\mathbf{P}}_t \in \mathfrak{R}^{m \cdot N \times 1}$, which captures the probability of each node being in any of m states at a given time t . Specifically:

$$\tilde{\mathbf{P}}_t = [P_{s_1,1,t}, P_{s_1,2,t}, \dots, P_{s_1,N,t}, P_{s_2,1,t}, \dots, P_{s_m,N,t}]^T \quad (2.3)$$

where, $P_{s_i,j,t}$ is the probability that node j is in state s_i at time t . A Non-Linear Dynamical System (NLDS) can be represented by $\tilde{\mathbf{P}}_{t+1} = g(\tilde{\mathbf{P}}_t)$ where g is some non-linear function operating on a vector. The function g in our case is large and complicated. The NLDS equation essentially tracks the evolution of the vector $\tilde{\mathbf{P}}_t$ over time. An equilibrium point (also called a fixed point) of the system is the state vector (i.e. some particular $\tilde{\mathbf{P}}$) which does not change. Thus at the equilibrium point $\tilde{\mathbf{P}}_{t+1} = \tilde{\mathbf{P}}_t = \tilde{\mathbf{x}}$. Intuitively, the tipping point for any model then deals with analyzing the stability of the corresponding NLDS at the point when none of the nodes in the graph are infected, because otherwise the infection can still spread. If the equilibrium is unstable, a small “perturbation” (physically in the form of a few initial nodes getting infected) will push the system further away (which physically means more and more nodes will get infected leading to an epidemic). But if the equilibrium is stable, the system will try to come back to the fixed point without going “too-far” away, in effect, “controlling the damage”. At threshold, the tendencies to go further away and come-back will be the same. In other words, the equilibrium is stable below the threshold and is neutral at the tipping point. From dynamical-system

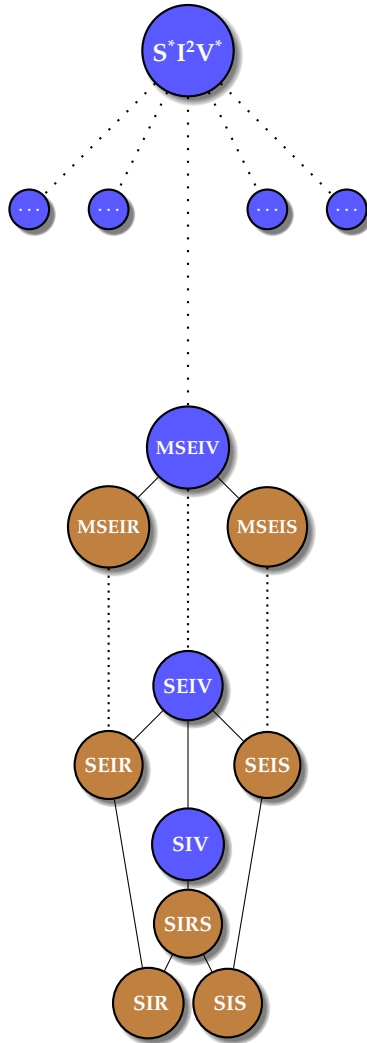


Figure 2.3: Virus Propagation model hierarchy (actually, lattice) for some standard models including SIRS (temporary immunity), SIV (vigilance, i.e., pro-active vaccination); SEIV (includes the ‘exposed but not infectious’ state, and temporary vigilance); MSEIR (with the passive immune state M); and our main generalization $S^*I^2V^*$. The brown colored nodes denote standard VPMs found in literature while the blue colored nodes denote our generalizations. Each VPM is a generalization of all the models below it.

literature, we know how to relate the stability of the system at the equilibrium point to the spectrum of the Jacobian matrix at that point (i.e. $\nabla g(\tilde{\mathbf{x}})$). We eventually reduced the requirement on the eigenvalues of $\nabla g(\tilde{\mathbf{x}})$ for any virus propagation model to a simple condition on the eigenvalue of the adjacency matrix. This condition translates into the effective strength of the virus under the model. The reason we can reduce the condition to one on the adjacency matrix is due to the special structure of the virus models, which was captured by the $S^*I^2V^*$ model described before. See the Appendix for the full proof.

2.6 Experiments

We performed computer simulation experiments on two large networks topologies, to demonstrate our result. All the different virus propagation models were implemented as a discrete event simulation in C++. We ran each simulation for 1000 time ticks and took the average of 100 runs. Initially, 10 nodes were infected with the virus and we then let the propagation take over according to the particular model. The datasets we used were:

1. AS-OREGON: This network represents the Internet’s Autonomous System (AS) connectivity derived from public data sets collected by the Oregon Route Views project¹. It contains 15,420 links among 3,995 AS peers. The Oregon graph is relevant to studying the robustness of router networks to worm attacks [LZZ⁺03]. More information can be found from <http://topology.eecs.umich.edu/data.html>.
2. PORTLAND: It is one of the biggest available physical contact graphs, representing a synthetic population of the city of Portland, Oregon, USA [NDS07]. It is a social-contact graph containing more than 31 mil. links (interactions) among about 1.6 mil. nodes (people). The data set is based on detailed microscopic simulation-based modeling and integration techniques and has been used in modeling studies on smallpox outbreaks as well as policy making at the national level [EGAK⁺04].

Figures 2.4 and 2.5 illustrate our result via simulation experiments on PORTLAND and AS-OREGON respectively. Above threshold, note the steady-state behavior in SEIV and the initial explosive phase and eventual decay in SIR and SEIR (because the number of susceptible nodes decrease monotonically). Also note the initial ‘flat’ period in the time plots for above threshold for the models having the exposed (E) state, SEIR and SEIV. This is due to the virus incubation period because of which there is an initial delay in number of infected nodes. This then results in an initial ‘silent’ period after which the epidemic takes-off. As there is no such incubation period in SIR and SIRS, their plots do not show such silent periods.

In contrast, under threshold, the number of infections aggressively goes down to zero in all the models. In addition, as our result predicts, the precise point when the footprint of infection suddenly jumps in all models is at $s = 1$. The footprint measures the

¹The University of Oregon Route Views Project. <http://www.routeviews.org>

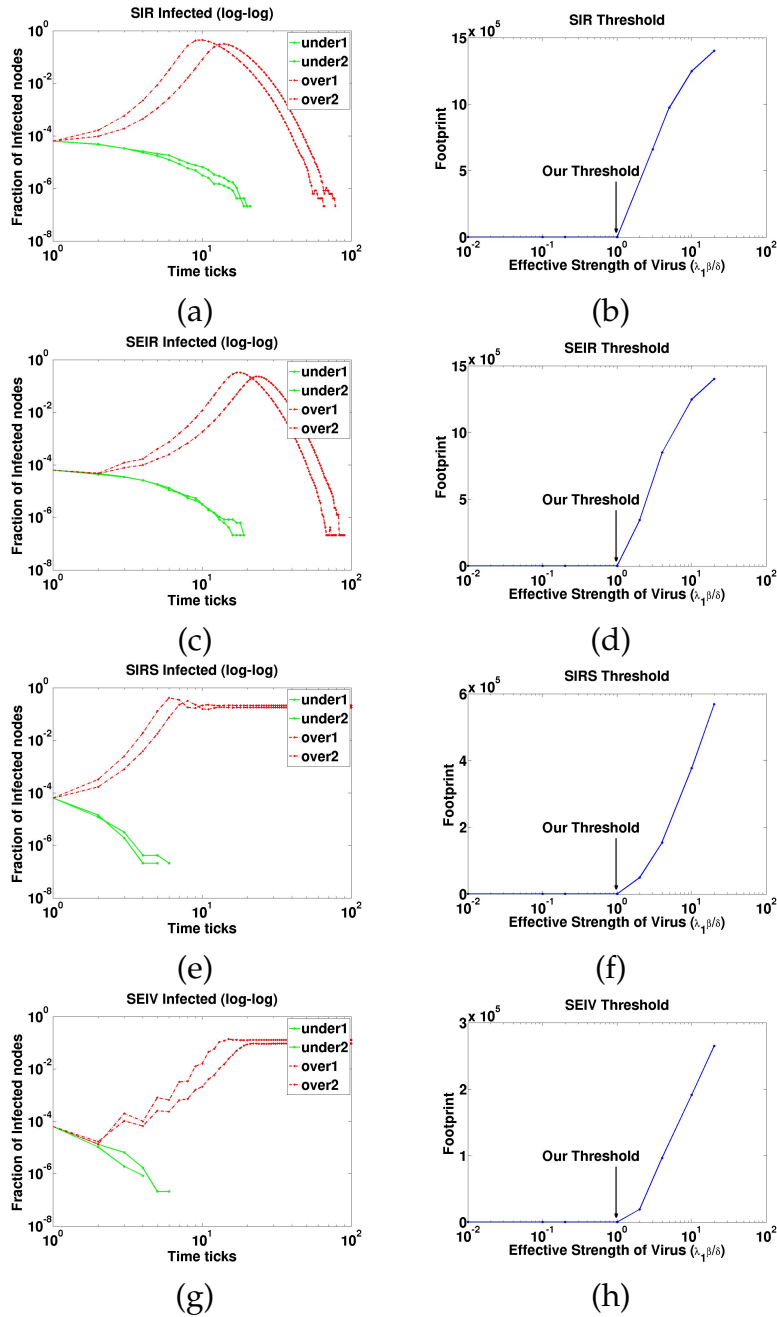


Figure 2.4: Simulation Results on the PORTLAND graph, all values averages over 100 runs. (a),(c),(e),(g) Plot of Infective Fraction of Population vs Time (log-log) for SIR, SEIR, SIRS and SEIV models. Note the qualitative difference in behavior- two curves *under* (green) the threshold and two curves *above* (red) the threshold. (b),(d),(f),(h) “Take-off” plots, *Footprint* (see Section 2.6) vs Effective Strength (lin-log) for SIR, SEIR, SIRS and SEIV models. The tipping point exactly matches our prediction ($s = 1$) in all cases.

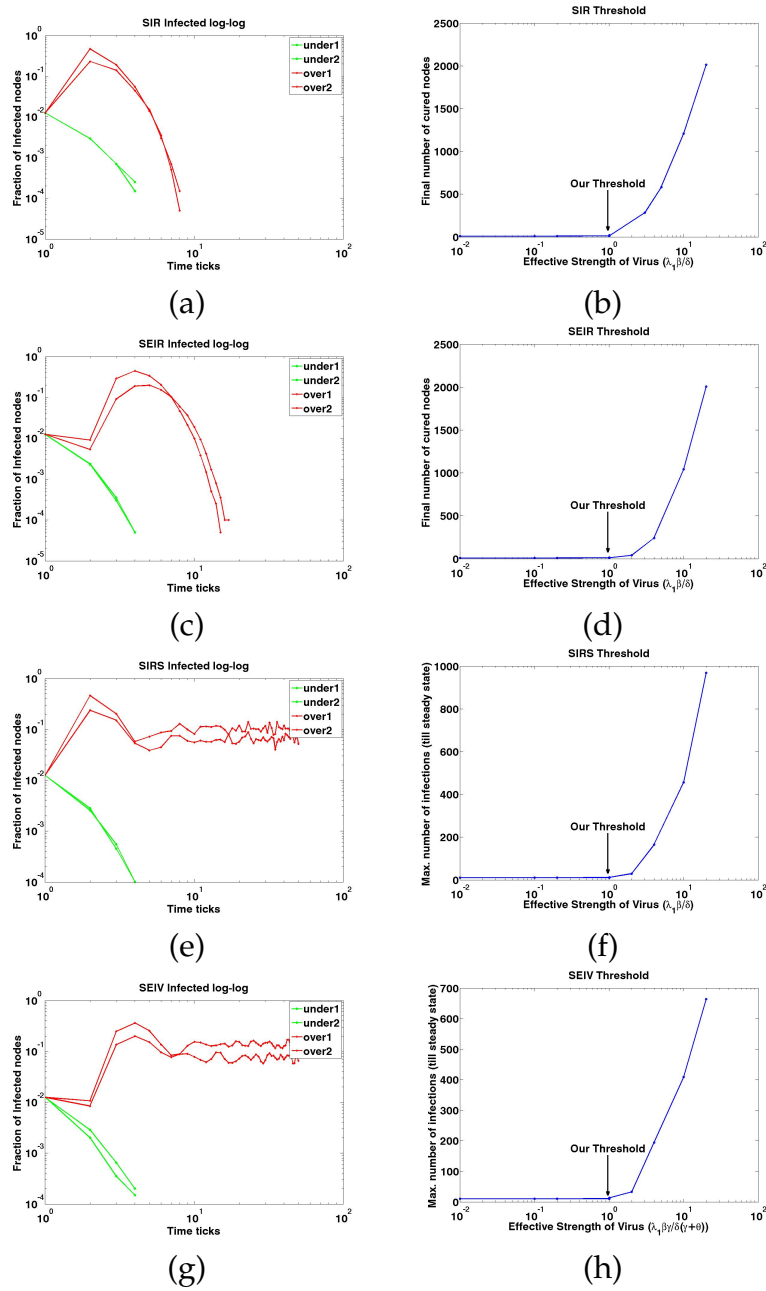


Figure 2.5: Simulation Results on the AS-OREGON graph, all values averages over 100 runs. **(a),(c),(e),(g)** Plot of Infective Fraction of Population vs Time (log-log) for SIR, SEIR, SIRS and SEIV models. Note the qualitative difference in behavior- two curves *under* (green) the threshold and two curves *above* (red) the threshold. **(b),(d),(f),(h)** "Take-off" plots, *Footprint* (see Section 2.6) vs Effective Strength (lin-log) for SIR, SEIR, SIRS and SEIV models. The tipping point exactly matches our prediction ($s = 1$) in all cases.

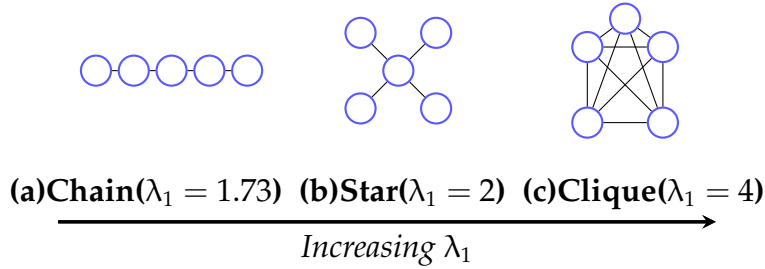


Figure 2.6: Why λ_1 matters more than number of edges E : changing connectivity and vulnerability of graphs with changing λ_1 . The clique (largest λ_1) is the most vulnerable. Note that E is not enough: star and chain have the same number of edges ($E = 4$) but the star is intuitively more vulnerable, as our result also says (it has a higher λ_1).

extent of infection: For models with a steady-state behavior (SIS/SIRS), it is defined as the maximum number of infections at any instant till we reach steady state. For models with monotonous decrease of susceptibles (and hence without a steady state, SIR/SEIR), footprint is the final number of cured/removed nodes from the network at the end of the infection. Figures 2.4 b, d, f, h and 2.5 b, d, f, h also demonstrate the simplicity and power of our result. the only variable we need for determining the epidemic threshold of the whole system consisting of multiple parameters is the effective strength ($s = \lambda_1 \cdot C_{VPM}$), nothing else.

2.7 Implications

We first discuss some direct implications of the G2-threshold theorem: the vulnerability of graphs to epidemics and some unexpected results in specific models.

2.7.1 Vulnerability of Networks—focus on eigenvalues

What exactly does the result mean w.r.t. the graph? Intuitively, λ_1 (also known as the spectral radius) of a graph captures the connectivity of the graph. More connected the graph is, more vulnerable it is to an epidemic by a virus (see Figure 2.6). Our threshold results suggest that an arbitrary graph behaves in the same way to a λ_1 -regular graph (both will have the same λ_1). The entire dynamics of the epidemic may not be captured by λ_1 completely, but the *threshold* is solely dependent on λ_1 (apart from parameters of the VPM). By making the relation between the graph and threshold explicit, our result has many consequences for the vulnerability of real, complex networks as well. For example, our result explains the observed vulnerability of ‘small-world’ networks [WS98]: their λ_1 is relatively high compared to a regular graph with the same number of nodes and edges, due to the presence of shortcuts. Also, previous results have shown that the epidemic threshold for the SIS model in case of random scale-free networks like the

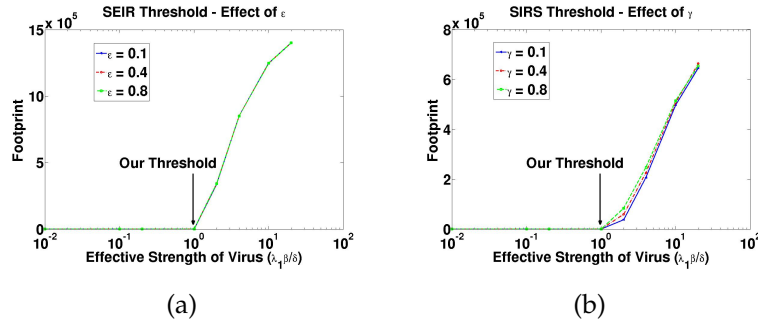


Figure 2.7: Counter-intuitive results - neither Incubation rate ϵ or Immunity-loss rate affects the threshold. **(a)** ‘Take-off’ plot for the SEIR model (a special case of SEIV) on the `PORTLAND` graph (lin-log scale). All three curves are on top of each other. **(b)** ‘Take-off’ plot for the SIRS model (a special case of SEIV) on the `PORTLAND` graph (lin-log scale) - higher means more infections (increasing with the loss of immunization γ). Note that in both the cases it *does not* affect the threshold (the tipping point is still at effective strength $s = 1$). All values are averages over 100 runs.

Internet is vanishingly small as the size N of the network increases [PSV01, AJB00]. This is a corollary of our result: When a power-law graph grows ($N \rightarrow \infty$), the largest eigenvalue grows with the maximum degree [CLV03], which also grows to infinity, and thus the threshold approaches zero.

2.7.2 Counter-intuitive Results

Apart from the dependence of the threshold on λ_1 , it is instructive to note unexpected results in some specific models. The SEIR, SIRS and SEIV models serve well to demonstrate the effect of virus-incubation and direct-immunization. See Figure 2.7. The threshold in SEIR surprisingly does not depend on the virus-incubation probability: the parameter ϵ , in effect, only delays/speeds-up the achievement of the threshold, not what the threshold itself is. Similarly, the threshold in SIRS does not depend on γ . Also, from the threshold equation of SEIV (Table 2.3), we can infer that lowering the rate of loss of immunity i.e. having a smaller γ (say due to better hygiene) decreases the effective strength s (and makes it harder for the virus to cause an epidemic) only so long as there is a mechanism to give a node direct immunity i.e. having a non-zero θ (say by using a vaccine) *before* an infection (in the Susceptible state) instead of *after* (in the Recovered state). Satisfyingly, this fits well with the old adage ‘Prevention is better than Cure’.

2.8 Impact

Our results can be fundamental to a wide-range of applications. We mentioned broader impact in § 2.1 before. Here we briefly discuss some immediate applications in epidemi-

ology and cyber-physical infrastructures.

2.8.1 Effective Immunization

Given the linear dependence on λ_1 of our threshold, we can propose a simple immunization goal. For any virus, remove (immunize) those nodes whose removal will decrease the λ_1 value the most (so that the resultant infection falls below threshold and dies out) e.g., immunize teachers and kindergarten children first to control the epidemic. A lot of work targets immunizing high-degree nodes in scale-free networks [CHbA03] which, while a good idea, is not optimal: just concentrating on high-degree nodes will *miss* those low-degree nodes which are good “bridges” and can have an important influence on decreasing λ_1 when immunized. For example, intuitively, the *sole common* friend between two disparate yet internally well-connected groups (like say between scientists and movie celebrities) can have a huge impact in the outbreak of a disease even if (s)he knows only a few people in each community.

2.8.2 Evaluating ‘What-if’ Scenarios

Our result can also help quickly determine the result of plausible situations e.g., is there a danger of an epidemic if the virus is twice (or half) as infectious (virulent)? This can then feed into policy decisions for controlling epidemics, like imposing restrictions on travel so as to not increase the λ_1 . Policy makers can assume *any* graph model which best captures the contact behavior of the population and still use our threshold result to guide immunization policies.

2.8.3 Accelerating Simulations

Similarly, we can considerably simplify expensive epidemiological simulations as well. For example, running a typical simulation with one set of parameters of a flu epidemic on a population of size 33 million (\sim size of the state of California) takes about 2 *days* on a cluster of 50 machines [BBE⁺08]. Using our result, we can eliminate parameters which do not affect the effective strength of the contagion and also quickly identify parameter spaces where simulations would be useful (i.e. above threshold). Clearly, the main task of such a testing will be eigenvalue computations. For this purpose, there are already very efficient algorithms like Lanczos for sparse graphs which take 2-5 *mins* for networks of millions. Moreover, structured topologies like cliques, block-diagonal matrices lend themselves to even faster eigenvalue computations, making it very easy to apply our result to real world simulations.

2.8.4 Applications to Computer Networking

As mentioned in Section 2.2, the epidemic threshold can be applied to a number of network robustness scenarios in cyber-physical infrastructures. For example, the Kademlia DHT [MM02] is used in a number of P2P networks—such as BitTorrent—to form a decentralized P2P overlay and lookup table. In particular, Mainline BitTorrent [(We11)] implements a version of Kademlia as an alternative to the typical centralized tracker. When a BitTorrent user queries the system for a particular torrent file, it is the DHT’s responsibility to return the torrent file (i.e., BitTorrent swarm bootstrap information). Hence, using our result, the network overlay structure of the DHT—in particular, its eigenvalue—may be used to evaluate the data replication necessary to guarantee the torrent file is reachable.

2.9 Conclusion

In summary, we studied the problem of determining the epidemic threshold given the virus propagation model and an underlying arbitrary undirected unweighted graph. Intuitively, the answer should depend both on the graph and the propagation model. Earlier results have focused on either special cases of graphs or special models. In this chapter, we give a formula for the epidemic threshold which shows:

1. *De-coupling*: The effect of the topology and the propagation model on the threshold is clearly de-coupled,
2. *Arbitrary Topology*: The effect of the undirected underlying topology is determined *only* by λ_1 (the largest eigenvalue of the adjacency matrix),
3. *Arbitrary VPM*: The effect of the virus propagation model is determined by a model dependent constant.

Thus, all previous epidemic threshold results are specific instantiations of our G_2 -threshold theorem. Our results can be used for forecasting and estimations in ‘what-if’ scenarios, for control and manipulation of propagation and related dynamical processes (immunization, marketing policies etc.). Moreover, our result can be easily extended to handle even more elaborate settings such as (a) time-varying topologies (extending the SIS-only results of [BBE⁺08, PTV⁺10]), and (b) multiple competing diseases (extending the random power-law-graphs-only results of [New05a]).

APPENDIX

We give the full proof of Theorem 2.1 here.

2.A Notation

Recall that we are dealing with the $S^*I^2V^*$ generalized model - it has two states I_1 and I_2 in the Infected class. To simplify notation, we refer to state I_1 as E (the ‘infection entrance state’) and I_2 as the I state in the proofs. The state E has a transmission probability of β_1 and the state I has a transmission probability of β_2 . The states E and I here should be thought as to mean *general* infected states of our model and not in the sense of the specific E and I states in epidemic models like SEIR, SEIV etc. We also refer to the exogenous transitions as *graph-based* and endogenous transitions as *internal* interchangeably. Table 2.4 gives some of the additional notation we will be using in our description of the proof.

Table 2.4: Additional Notation and Symbols used in the proofs

Symbol	Definition
m	total number of states in the model
q	total number of states in the Susceptible and Vigilant classes of the model; hence $m = q + 2$
w	total number of states in the Susceptible class of the model
S_1, S_2, \dots, S_w	general states in the Susceptible class
E, I	general states in the Infected class
α_{KU}	probability (constant and given) of transition from state K to state U
β_1	transmission probability for state E
β_2	transmission probability for state I
$\zeta_{i,t}(E, I)$	probability that a node i does not receive any infections from E and I at time t
\tilde{x}	the fixed point vector our NLDS corresponding to when no node is in any of the Infected class states
$p_{S_y}^*$	(same for each node) probability of being present in the S_y state at \tilde{x}
\mathcal{J}	Jacobian matrix of the NLDS computed at \tilde{x}

2.B System Equations

We can develop the system equations i.e. explicitly specify the non-linear function g for the NLDS based on the transition diagram of the model. As stated earlier in Section 2.5.2 we assume that infections are received only from infected neighbors i.e. those in states E and I the Infected class of states. Firstly, let’s calculate the probability that a node i does not receive any infections in the next time step (call it $\zeta_{i,t}(E, I)$, E, I denotes that an infection is passed only from a neighbor in the E or I states). No infections are transmitted if:

- Either a neighbor is not any of the infected states E and I

- Or it is in state E and the transmission fails with probability $1 - \beta_1$
- Or it is in state I and the transmission fails with probability $1 - \beta_2$

Since we assume infinitesimally small time steps ($\Delta t \rightarrow 0$), multiple events can be ignored for first-order effects in the time step. Also, assuming the neighbors are *independent*, we get:

$$\begin{aligned}\zeta_{i,t}(E, I) &= \prod_{j \in \mathcal{NE}(i)} (P_{E,j,t}(1 - \beta_1) + P_{I,j,t}(1 - \beta_2) + (1 - P_{E,j,t} - P_{I,j,t})) \\ &= \prod_{j \in \{1..N\}} (1 - \mathbf{A}_{i,j}(\beta_1 P_{E,j,t} + \beta_2 P_{I,j,t}))\end{aligned}\quad (2.4)$$

where $\mathcal{NE}(i)$ is the set of neighbors of node i in the graph.

Also, the sum of probabilities of being in all the possible states for each node i should equal 1. Hence,

$$\forall_{i,t} \sum_K P_{K,i,t} = 1 \quad (2.5)$$

We can now write down the system equations as follows. A node i will be in any particular state S_y of the Susceptible class at time $t + 1$ if:

- Either it was in S_y at time t and stayed in state S_y i.e. it did not receive any infections from its neighbors *and* it did not change state internally from S_y to any other state
- Or it was in some other state U and changed state internally from U to S_y

Hence, the probability of node i being in S_y where S_y is any state in the Susceptible class at time $t + 1$ is:

$$\forall y = 1, 2, \dots, w \quad P_{S_y,i,t+1} = \sum_{K \neq S_y} \alpha_{KS_y} P_{K,i,t} + P_{S_y,i,t} \left(\zeta_{i,t}(E, I) - \sum_{K \neq E, S_y, I} \alpha_{S_y K} \right) \quad (2.6)$$

Similarly, for the E state:

$$P_{E,i,t+1} = \sum_{K \neq S_1, S_2, \dots, S_w} \alpha_{KE} P_{K,i,t} + \sum_{y=1}^w P_{S_y,i,t} (1 - \zeta_{i,t}(E, I)) \quad (2.7)$$

and for any other state $U \neq \{S_1, S_2, \dots, S_w, E\}$:

$$P_{U,i,t+1} = \sum_K \alpha_{KU} P_{K,i,t} \quad (2.8)$$

As discussed earlier (Equation 2.3), we can now define a probability vector $\tilde{\mathbf{P}}_t$ by “stacking” all these probabilities which will completely describe the system at any time t and evolve according to the above equations. Note that the above equations are non-linear and naturally define the function g for the NLDS $\tilde{\mathbf{P}}_{t+1} = g(\tilde{\mathbf{P}}_t)$.

We have the following theorem about NLDS stability at a fixed point:

Theorem 2.2 (Asymptotic Stability, e.g., see [HS74]). *The system given by $\tilde{\mathbf{P}}_{t+1} = g(\tilde{\mathbf{P}}_t)$ is asymptotically stable at an equilibrium point $\tilde{\mathbf{P}} = \tilde{\mathbf{x}}$, if the eigenvalues of $\mathcal{J} = \nabla g(\tilde{\mathbf{x}})$ are less than 1 in absolute value, where,*

$$\mathcal{J}_{i,j} = [\nabla g(\tilde{\mathbf{x}})]_{i,j} = \left. \frac{\partial g_i}{\partial g_j} \right|_{\tilde{\mathbf{P}}=\tilde{\mathbf{x}}}$$

Hence, next we compute the fixed point we are interested in and the Jacobian of our NLDS at that point.

2.C Fixed point

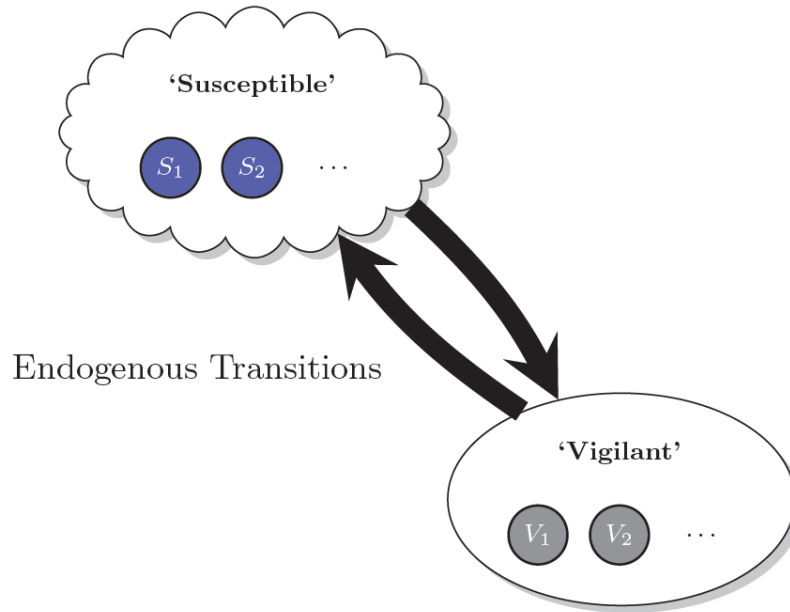


Figure 2.8: State Diagram for any node in the graph at the fixed point when *no* node is present in a state in the Infected class. Only cross-class edges are shown. Note that it is now a simple Markov chain with a unique steady state probability.

We are interested in the stability of the equilibrium point (i.e. where $\tilde{\mathbf{P}}_{t+1} = \tilde{\mathbf{P}}_t (= \tilde{\mathbf{x}})$) of the NLDS which corresponds to when no one is infected. Only the transition from the Susceptible class states towards the Infected class states are graph-based (and can happen only when at least one of the nodes is in any of the Infected states), so the state-diagram for each node will be a simple Markov chain (call it MC_{SV}) consisting of the Susceptible and Vigilant states (see Figure 2.8). Note now there are no graph-based effects, hence each node is independent of others and will converge to steady state probabilities corresponding to the Markov chain. The steady state vector π^* (size $q \times 1$, where q is the number of states in the Susceptible and Vigilant classes) which will be

the same for each node can be computed from the following equations from standard Markov chain analysis:

$$\pi^{*\top} \text{Tran}_{\text{MC}_{SV}} = \pi^* \quad \& \quad \sum_{i=1}^q \pi_i^* = 1 \quad (2.9)$$

Hence π^* is a probability vector and is the left eigenvector corresponding to eigenvalue 1 of the stochastic matrix $\text{Tran}_{\text{MC}_{SV}}$ of the Markov chain MC_{SV} . The full $(m \times 1)$ probability vector $\tilde{\mathbf{p}}^*$ for each node at this steady state will have the entries in π^* for states in the Susceptible and Vigilant classes and 0 for all states in the Infected class. The fixed point of the global original NLDS $\tilde{\mathbf{x}}$ can be finally represented as:

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{\mathbf{p}}^* \\ \tilde{\mathbf{p}}^* \\ \vdots \\ \tilde{\mathbf{p}}^* \end{bmatrix} \quad (2.10)$$

where $\tilde{\mathbf{p}}^*$ is repeated N times (once for each node in the graph). Let $p_{S_y}^*$ be the steady state probability value in the vector $\tilde{\mathbf{p}}^*$ corresponding to the S_y state. In other words, each node will have a probability of $p_{S_y}^*$ of being present in the S_y state at the fixed point. Also define,

$$p_S^* = \sum_{y=1}^w p_{S_y}^* \quad (2.11)$$

i.e. p_S^* is the total probability of each node at the fixed point of being present in any of the states of the Susceptible class.

2.D The Jacobian

We know from Theorem 2.2 that $\tilde{\mathbf{x}}$ is stable if the eigenvalues of $\mathcal{J} = \nabla g(\tilde{\mathbf{x}})$ are less than 1 in absolute value. From the definition of \mathcal{J} we can see that it is a $m \cdot N \times m \cdot N$ matrix with m (for each state) square blocks of size $N \times N$ each (corresponding to every node in the graph). We can calculate \mathcal{J} to be (states have been mentioned on the top and side for ease of exposition and \mathbf{I} is the identity matrix of size $N \times N$):

	S_y	K	...	E	I
S_y	$(1 - \sum_{K \neq S_y, E} \alpha_{S_y, K}) \mathbf{I}$	$\alpha_{KS_y} \mathbf{I}$...	$\alpha_{ES_y} \mathbf{I} - p_{S_y}^* \beta_1 \mathbf{A}$	$\alpha_{IS_y} \mathbf{I} - p_S^* \beta_2 \mathbf{A}$
\vdots			\ddots		
U	$\alpha_{S_y U} \mathbf{I}$	$\alpha_{KU} \mathbf{I}$...	$\alpha_{EU} \mathbf{I}$	$\alpha_{IU} \mathbf{I}$
\vdots			\ddots		
E	$\alpha_{S_y E} \mathbf{I}$	$\alpha_{KE} \mathbf{I}$...	$\alpha_{EE} \mathbf{I} + p_S^* \beta_1 \mathbf{A}$	$\alpha_{IE} \mathbf{I} + p_S^* \beta_2 \mathbf{A}$
I	$\alpha_{S_y I} \mathbf{I}$	$\alpha_{KI} \mathbf{I}$...	$\alpha_{EI} \mathbf{I}$	$\alpha_{II} \mathbf{I}$

where K is any state $\neq \{E, I\}$ and U is any state $\neq \{S_1, S_2, \dots, S_w, E, I\}$.

Recall the properties we are assuming for the epidemic models discussed in Section 2.5.2 (also see Figure 2.2). Crucially, they imply $\forall_{K \neq E, I} \alpha_{KE} = 0$ and $\forall_{K \neq E, I} \alpha_{KI} = 0$. Hence \mathcal{J} reduces to:

	S_y	K	...	E	I
S_y	$(1 - \sum_{K \neq S_y, E} \alpha_{S_y, K}) \mathbf{I}$	$\alpha_{KS_y} \mathbf{I}$...	$\alpha_{ES_y} \mathbf{I} - p_{S_y}^* \beta_1 \mathbf{A}$	$\alpha_{IS_y} \mathbf{I} - p_S^* \beta_2 \mathbf{A}$
\vdots			\ddots		
U	$\alpha_{S_y U} \mathbf{I}$	$\alpha_{KU} \mathbf{I}$...	$\alpha_{EU} \mathbf{I}$	$\alpha_{IU} \mathbf{I}$
\vdots			\ddots		
E	$\mathbf{0}_{N, N}$	$\mathbf{0}_{N, N}$...	$\alpha_{EE} \mathbf{I} + p_S^* \beta_1 \mathbf{A}$	$\alpha_{IE} \mathbf{I} + p_S^* \beta_2 \mathbf{A}$
I	$\mathbf{0}_{N, N}$	$\mathbf{0}_{N, N}$...	$\alpha_{EI} \mathbf{I}$	$\alpha_{II} \mathbf{I}$

where $\mathbf{0}_{N, N}$ is a $N \times N$ matrix with all zeros.

2.E Eigenvalues of the Jacobian

Note that \mathcal{J} is very structured and can be written as:

$$\mathcal{J} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{0}_{2N, (m-2)N} & \mathbf{B}_3 \end{bmatrix} \quad (2.12)$$

where \mathbf{B}_1 , \mathbf{B}_2 and \mathbf{B}_3 are matrices of size $(m-2)N \times (m-2)N$, $(m-2)N \times 2N$ and $2N \times 2N$ respectively. \mathbf{B}_3 corresponds to the E and I rows and columns of \mathcal{J} i.e.:

$$\mathbf{B}_3 = \begin{bmatrix} \alpha_{EE} \mathbf{I} + p_S^* \beta_1 \mathbf{A} & \alpha_{IE} \mathbf{I} + p_S^* \beta_2 \mathbf{A} \\ \alpha_{EI} \mathbf{I} & \alpha_{II} \mathbf{I} \end{bmatrix} \quad (2.13)$$

\mathbf{B}_1 and \mathbf{B}_2 are defined similarly. Consider any eigenvector $\tilde{\mathbf{v}}$ (size $mN \times 1$) and corresponding eigenvalue λ_j of \mathcal{J} . We can write $\tilde{\mathbf{v}}$ as being composed of vector $\tilde{\mathbf{v}}_1$ of size $(m-2)N \times 1$ and vector $\tilde{\mathbf{v}}_2$ of size $2N \times 1$ i.e:

$$\tilde{\mathbf{v}} = \begin{bmatrix} \tilde{\mathbf{v}}_1 \\ \tilde{\mathbf{v}}_2 \end{bmatrix} \quad (2.14)$$

Also $\tilde{\mathbf{x}}$ and λ_j satisfy the eigenvalue equation:

$$\mathcal{J}\tilde{\mathbf{v}} = \lambda_j\tilde{\mathbf{v}} \quad (2.15)$$

Substituting from Equations 2.12 and 2.14 we get:

$$\begin{bmatrix} \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{0} & \mathbf{B}_3 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{v}}_1 \\ \tilde{\mathbf{v}}_2 \end{bmatrix} = \lambda_j \begin{bmatrix} \tilde{\mathbf{v}}_1 \\ \tilde{\mathbf{v}}_2 \end{bmatrix} \quad (2.16)$$

Equation 2.16 implies the following the two relations:

$$\mathbf{B}_1\tilde{\mathbf{v}}_1 + \mathbf{B}_2\tilde{\mathbf{v}}_2 = \lambda_j\tilde{\mathbf{v}}_1 \quad (2.17)$$

$$\mathbf{B}_3\tilde{\mathbf{v}}_2 = \lambda_j\tilde{\mathbf{v}}_2 \quad (2.18)$$

From Equation 2.18 we can infer that precisely *one* of the following holds:

1. $\tilde{\mathbf{v}}_2 = \tilde{\mathbf{0}}$
2. $\tilde{\mathbf{v}}_2$ is the eigenvector of \mathbf{B}_3 (and consequently λ_j is the matching eigenvalue of \mathbf{B}_3)

If $\tilde{\mathbf{v}}_2 = \tilde{\mathbf{0}}$, Equation 2.17 reduces to

$$\mathbf{B}_1\tilde{\mathbf{v}}_1 = \lambda_j\tilde{\mathbf{v}}_1$$

wherein again, either $\tilde{\mathbf{v}}_1 = \tilde{\mathbf{0}}$ or λ_j is an eigenvalue of \mathbf{B}_1 . The condition $\tilde{\mathbf{v}}_1 = \tilde{\mathbf{0}}$ is not meaningful as then $\tilde{\mathbf{v}} = \tilde{\mathbf{0}}$ ($\tilde{\mathbf{v}}$ is an eigenvector of \mathcal{J} implies $\tilde{\mathbf{v}}$ is non-zero). Therefore the eigenvalues of \mathcal{J} are given by the eigenvalues of \mathbf{B}_1 (with $\tilde{\mathbf{v}}_2 = \tilde{\mathbf{0}}$) and the eigenvalues of \mathbf{B}_3 .

2.E.1 Eigenvalues of \mathbf{B}_1

From the expression for \mathcal{J} derived in Section 2.D, note that:

$$\mathbf{B}_1 = \mathbf{T} \otimes \mathbf{I} \quad (2.19)$$

where \otimes is the Kronecker product of two matrices and

$$\mathbf{T} = \begin{bmatrix} (1 - \sum_{\kappa \neq S_y, E} \alpha_{S_y \kappa}) & \alpha_{KS_y} & \dots \\ \vdots & \vdots & \vdots \\ \alpha_{S_y U} & \alpha_{KU} & \dots \\ \vdots & \ddots & \vdots \end{bmatrix} \quad (2.20)$$

We know from matrix algebra [HJ91] that if $\mathbf{C} = \mathbf{D} \otimes \mathbf{E}$ then $\mathbf{C}_\lambda = \mathbf{D}_\lambda \otimes \mathbf{E}_\lambda$, where \mathbf{C}_λ denotes a diagonal matrix with eigenvalues of the matrix \mathbf{C} on the diagonal. But $\mathbf{I}_\lambda = \mathbf{I}$, hence the eigenvalues of \mathbf{B}_1 are the same as the eigenvalues of \mathbf{T} (although with repetition). In other words, eigenvalues of \mathbf{T} are eigenvalues of \mathcal{J} as well.

2.E.2 Eigenvalues of \mathbf{B}_3

Let $\tilde{\mathbf{u}} = \begin{bmatrix} \tilde{\mathbf{u}}_1 \\ \tilde{\mathbf{u}}_2 \end{bmatrix}$ be a corresponding eigenvector of \mathbf{B}_3 ($\tilde{\mathbf{u}}_1$ and $\tilde{\mathbf{u}}_2$ are of size $N \times 1$ each and as the eigenvalues of \mathbf{B}_3 are also eigenvalues of \mathcal{J} , we use $\lambda_{\mathcal{J}}$ for an eigenvalue of \mathbf{B}_3). Hence, the standard eigenvalue relation $\mathbf{B}_3 \tilde{\mathbf{u}} = \lambda_{\mathcal{J}} \tilde{\mathbf{u}}$ requires the following equations to be satisfied:

$$(\alpha_{EE} \mathbf{I} + p_S^* \beta_1 \mathbf{A}) \tilde{\mathbf{u}}_1 + (\alpha_{IE} \mathbf{I} + p_S^* \beta_2 \mathbf{A}) \tilde{\mathbf{u}}_2 = \lambda_{\mathcal{J}} \tilde{\mathbf{u}}_1 \quad (2.21)$$

$$\alpha_{EI} \tilde{\mathbf{u}}_1 + \alpha_{II} \tilde{\mathbf{u}}_2 = \lambda_{\mathcal{J}} \tilde{\mathbf{u}}_2 \quad (2.22)$$

Using Equation 2.22, we can compute $\tilde{\mathbf{u}}_1$ in terms of $\tilde{\mathbf{u}}_2$ as:

$$\tilde{\mathbf{u}}_1 = \left(\frac{\lambda_{\mathcal{J}} - \alpha_{II}}{\alpha_{EI}} \right) \tilde{\mathbf{u}}_2 \quad (2.23)$$

Substituting it back into Equation 2.21 we get:

$$\begin{aligned} & \left((\alpha_{EE} \mathbf{I} + p_S^* \beta_1 \mathbf{A}) \left(\frac{\lambda_{\mathcal{J}} - \alpha_{II}}{\alpha_{EI}} \right) + \alpha_{IE} \mathbf{I} + p_S^* \beta_2 \mathbf{A} \right) \tilde{\mathbf{u}}_2 = \lambda_{\mathcal{J}} \left(\frac{\lambda_{\mathcal{J}} - \alpha_{II}}{\alpha_{EI}} \right) \tilde{\mathbf{u}}_2 \\ \Rightarrow & (\alpha_{EE}(\lambda_{\mathcal{J}} - \alpha_{II}) \mathbf{I} + \alpha_{IE} \alpha_{EI} \mathbf{I} + (p_S^* \beta_1(\lambda_{\mathcal{J}} - \alpha_{II}) + p_S^* \beta_2 \alpha_{EI}) \mathbf{A}) \tilde{\mathbf{u}}_2 = \lambda_{\mathcal{J}}(\lambda_{\mathcal{J}} - \alpha_{II}) \tilde{\mathbf{u}}_2 \end{aligned}$$

which finally gives,

$$\mathbf{A} \tilde{\mathbf{u}}_2 = \left(\frac{\lambda_{\mathcal{J}}^2 - (\alpha_{II} + \alpha_{EE}) \lambda_{\mathcal{J}} + \alpha_{II} \alpha_{EE} - \alpha_{IE} \alpha_{EI}}{p_S^* \beta_1 (\lambda_{\mathcal{J}} - \alpha_{II}) + p_S^* \beta_2 \alpha_{EI}} \right) \tilde{\mathbf{u}}_2 \quad (2.24)$$

Again, Equation 2.24 tells us that either $\tilde{\mathbf{u}}_2 = \tilde{\mathbf{0}}$ or it is an eigenvector for \mathbf{A} . But $\tilde{\mathbf{u}}_2 = \tilde{\mathbf{0}} \Rightarrow \tilde{\mathbf{u}}_1 = \tilde{\mathbf{0}} \Rightarrow \tilde{\mathbf{u}} = \tilde{\mathbf{0}}$ which is not possible. Thus Equation 2.24 is an eigenvalue equation for the adjacency matrix \mathbf{A} and we are looking for solutions $\lambda_{\mathcal{J}}$ and $\tilde{\mathbf{u}}_2$ such that they satisfy it. Hence,

$$\lambda_{\mathbf{A}} = \frac{\lambda_{\mathcal{J}}^2 - (\alpha_{II} + \alpha_{EE}) \lambda_{\mathcal{J}} + \alpha_{II} \alpha_{EE} - \alpha_{IE} \alpha_{EI}}{p_S^* \beta_1 (\lambda_{\mathcal{J}} - \alpha_{II}) + p_S^* \beta_2 \alpha_{EI}}$$

where $\lambda_{\mathbf{A}}$ is an eigenvalue of \mathbf{A} . This finally gives

$$\lambda_{\mathcal{J}}^2 - \lambda_{\mathcal{J}}(\alpha_{EE} + \alpha_{II} + p_S^* \beta_1 \lambda_{\mathbf{A}}) + (\alpha_{II} \alpha_{EE} - \alpha_{IE} \alpha_{EI} + p_S^* \lambda_{\mathbf{A}} (\beta_1 \alpha_{II} - \beta_2 \alpha_{EI})) = 0 \quad (2.25)$$

Thus we have a different quadratic equation (Q.E.) for each eigenvalue $\lambda_{\mathbf{A}}$ of \mathbf{A} . Each Q.E. gives us two eigenvalues (possibly repeated) of \mathcal{J} .

So, finally, we can conclude the following lemma:

Lemma 2.1 (Eigenvalues of \mathcal{J}). *Eigenvalues of \mathcal{J} are given by the eigenvalues of \mathbf{T} (Equations 2.19 and 2.20) and the roots of the Q.Es given by Equation 2.25 for each eigenvalue $\lambda_{\mathbf{A}}$ of \mathbf{A} .*

2.F Stability

We require that all the eigenvalues of \mathcal{J} to be less than 1 in absolute value (according to Theorem 2.2). From Lemma 2.1, we have two cases to handle in enforcing this:

- (C1) All the eigenvalues of \mathbf{T} should be less than 1 in absolute value
- (C2) All the roots of the Q.Es given by Equation 2.25 for each eigenvalue λ_A of \mathbf{A} should be less than 1 in absolute value

2.F.1 Case C1

Note that this case depends *only* on the model as the matrix \mathbf{T} is *independent* of the adjacency matrix \mathbf{A} . But \mathbf{T} is a stochastic matrix i.e. all the column sums are equal to 1 - consequently all its eigenvalues are less than 1 in absolute value.

Lemma 2.2 (Stability C1). *All eigenvalues of the matrix \mathbf{T} (given by Equation 2.20) are less than 1 in absolute value.*

2.F.2 Case C2

As C1 is always true, we need to only ensure case C2. We can prove here the following:

Lemma 2.3 (Stability C2). *All the roots of the Q.Es given by Equation 2.25 for each eigenvalue λ_A of \mathbf{A} are less than 1 in absolute value if:*

$$\lambda_1 p_s^* \left(\frac{\beta_1(1 - \alpha_{II}) + \beta_2 \alpha_{EI}}{(1 - \alpha_{II})(1 - \alpha_{EE}) - \alpha_{IE} \alpha_{EI}} \right) < 1$$

Proof Let r_1 and r_2 be the roots of Equation 2.25 (r_1 and r_2 can be real or complex depending on λ_A). Then we want

$$|r_1| < 1 \text{ and } |r_2| < 1$$

r_1 and r_2 are real As the roots are real, λ_A is such that the discriminant \mathcal{D} of the quadratic equation is greater than zero. In this situation:

$$|r_1| < 1 \text{ and } |r_2| < 1 \Rightarrow r_1 \in (-1, 1) \text{ and } r_2 \in (-1, 1) \quad (2.26)$$

From the theory of quadratic equations, it is well known (see e.g., [Mil63]) that for real roots x_1 and x_2 of a Q.E. $f(x) = ax^2 + bx + c$ (with $a > 0$) to lie in the interval $(-1, 1)$ the following conditions must be true:

$$\begin{aligned} a - c &> 0, \\ a - b + c &> 0, \\ a + b + c &> 0. \end{aligned}$$

Intuitively, the first condition forces the product of the roots to be less than 1 while the last two conditions state that value of $f(x)$ at -1 and 1 should not be “too small”. In our case, these then translate into:

$$\alpha_{II}\alpha_{EE} - \alpha_{IE}\alpha_{EI} + p_S^*\lambda_A(\beta_1\alpha_{II} - \beta_2\alpha_{EI}) < 1 \quad (2.27a)$$

$$1 + \alpha_{EE} + \alpha_{II} + p_S^*\beta_1\lambda_A + \alpha_{II}\alpha_{EE} - \alpha_{IE}\alpha_{EI} + p_S^*\lambda_A(\beta_1\alpha_{II} - \beta_2\alpha_{EI}) > 0 \quad (2.27b)$$

$$1 - \alpha_{EE} - \alpha_{II} - p_S^*\beta_1\lambda_A + \alpha_{II}\alpha_{EE} - \alpha_{IE}\alpha_{EI} + p_S^*\lambda_A(\beta_1\alpha_{II} - \beta_2\alpha_{EI}) > 0 \quad (2.27c)$$

Equations 2.27b and 2.27c can be written as:

$$\lambda_A p_S^* \left(\frac{-\beta_1(1 + \alpha_{II}) + \beta_2\alpha_{EI}}{(1 + \alpha_{II})(1 + \alpha_{EE}) - \alpha_{IE}\alpha_{EI}} \right) < 1 \quad (2.28a)$$

$$\lambda_A p_S^* \left(\frac{\beta_1(1 - \alpha_{II}) + \beta_2\alpha_{EI}}{(1 - \alpha_{II})(1 - \alpha_{EE}) - \alpha_{IE}\alpha_{EI}} \right) < 1 \quad (2.28b)$$

respectively. The above equations should be true for *any* eigenvalue λ_A of \mathbf{A} which makes $\mathcal{D} > 0$. Recall that we are considering only undirected graphs, hence \mathbf{A} is a symmetric binary (0/1) square irreducible matrix. As a result firstly, all its eigenvalues are real. Secondly, from the Perron-Frobenius theorem [McC00] the algebraically largest eigenvalue λ_1 of \mathbf{A} is a positive real number and also has the largest magnitude among all eigenvalues. Hence if the above equations are true for $\lambda_A = \lambda_1$ we are done. Now note that

$$(1 + \alpha_{II})(1 + \alpha_{EE}) - \alpha_{IE}\alpha_{EI} > (1 - \alpha_{II})(1 - \alpha_{EE}) - \alpha_{IE}\alpha_{EI}$$

and that

$$\beta_1(1 - \alpha_{II}) + \beta_2\alpha_{EI} > -\beta_1(1 + \alpha_{II}) + \beta_2\alpha_{EI}$$

In addition the L.H.S in both equations is positive under $\lambda_A = \lambda_1$. So Equation 2.28a is always true if Equation 2.28b holds (under $\lambda_A = \lambda_1$) i.e.

$$\lambda_1 p_S^* \left(\frac{\beta_1(1 - \alpha_{II}) + \beta_2\alpha_{EI}}{(1 - \alpha_{II})(1 - \alpha_{EE}) - \alpha_{IE}\alpha_{EI}} \right) < 1 \quad (2.29)$$

As λ_1 is the largest eigenvalue both algebraically and in magnitude, under Equation 2.29,

$$\begin{aligned} & 1 - \alpha_{II}\alpha_{EE} + \alpha_{IE}\alpha_{EI} - p_S^*\lambda_A(\beta_1\alpha_{II} - \beta_2\alpha_{EI}) \\ & > 1 - \alpha_{II}\alpha_{EE} + \alpha_{IE}\alpha_{EI} - \left(\frac{(1 - \alpha_{II})(1 - \alpha_{EE}) - \alpha_{IE}\alpha_{EI}}{\beta_1(1 - \alpha_{II}) + \beta_2\alpha_{EI}} \right) (\beta_1\alpha_{II} - \beta_2\alpha_{EI}) \\ & = \frac{((1 - \alpha_{II})^2 + \alpha_{IE}\alpha_{EI})\beta_1 + \alpha_{EI}(2 - \alpha_{II} - \alpha_{EE})\beta_2}{(1 - \alpha_{II})\beta_1 + \alpha_{EI}\beta_2} \\ & > 0 \end{aligned}$$

\therefore Equation 2.27a is also true if Equation 2.29 holds. Thus the condition for the roots to be in $(-1, 1)$ when they are real is given simply by Equation 2.29.

r_1 and r_2 are complex In this case λ_A is such that $\mathcal{D} < 0$. Also as Equation 2.25 has real co-efficients, r_1 and r_2 are complex conjugate of each other and so $|r_1| = |r_2| = \sqrt{r_1 \cdot r_2}$. But the product of roots x_1 and x_2 of the equation $ax^2 + bx + c = 0$ is equal to c/a . Hence we want to enforce $c/a < 1$. In our case it is

$$\alpha_{II}\alpha_{EE} - \alpha_{IE}\alpha_{EI} + p_S^*\lambda_A(\beta_1\alpha_{II} - \beta_2\alpha_{EI}) < 1$$

which is exactly Equation 2.27a. From the above analysis, we already know that it is true if Equation 2.29 holds. So, for any eigenvalue λ_A for which $\mathcal{D} < 0$, the roots have magnitude less than 1 given Equation 2.29 is true.

Thus in both cases, whether roots are real or complex, Equation 2.29 is a sufficient condition for the roots to have magnitude less than 1. \square

To re-cap we state our result and then give its proof:

Theorem 2.3 (G2 theorem). *For virus propagation models which satisfy our general initial assumptions and for any arbitrary undirected graph with adjacency matrix \mathbf{A} and largest eigenvalue λ_1 , the sufficient condition for stability is given by:*

$$s < 1$$

where, s (the effective strength) is:

$$s = \lambda_1 \cdot C$$

and C is a constant dependent on the model (given by Equation 2.30). Hence, the tipping point is reached when $s = 1$.

Proof Lemma 2.2 and Lemma 2.3 ensure cases C1 and C2 and hence together with Lemma 2.1 imply that the eigenvalues of the Jacobian \mathcal{J} of our general NLDS computed at the fixed point \tilde{x} are less than 1 in magnitude if Equation 2.29 is true.

\therefore using Theorem 2.2, our general NLDS is stable at its fixed point \tilde{x} if Equation 2.29 holds. Recall that \tilde{x} is the point when there no infected nodes in the system (Appendix 2.C) and that this is the fixed point whose stability conditions determine the epidemic threshold (Section 2.5).

\therefore finally we can conclude the theorem with

$$C_{VPM} = p_S^* \left(\frac{\beta_1(1 - \alpha_{II}) + \beta_2\alpha_{EI}}{(1 - \alpha_{II})(1 - \alpha_{EE}) - \alpha_{IE}\alpha_{EI}} \right) \quad (2.30)$$

and the effective strength $s = \lambda_1 \cdot C_{VPM}$. The parameter C_{VPM} is a constant for a given propagation model while the only parameter involved from the underlying contact-network is λ_1 , the first eigenvalue of the adjacency matrix. \square

Chapter 3

Epidemic Thresholds: Time-varying Graphs

In this chapter, we focus on contact networks that change over time (say, day vs night connectivity), and the SIS (susceptible/infected/susceptible, flu like) virus propagation model. Given such a configuration, we want to ask the same question as in the chapter before: what can we say about the epidemic threshold? That is, can we determine when a small infection will “take-off” and create an epidemic? This is a very real problem, since, for example, people have different connections during the day at work, and during the night at home. As we saw before, static graphs have been studied for a long time, with numerous analytical results. Time-evolving networks are so hard to analyze, that most existing works are simulation studies [BBE⁺08].

Specifically, our contributions in this chapter are: (a) we formulate the problem by approximating it by a Non-linear Dynamical system (NLDS), and (b) we derive the **first** closed formula for the epidemic threshold of time-varying graphs under the SIS model.

3.1 Introduction

The goal of this work is to analytically study the epidemic spread on time-varying graphs. We focus on time-varying graphs that follow an alternating connectivity behavior, which is motivated by the day-night pattern of human behavior. Note that our analysis is not restricted to two graphs: we can have an arbitrary number of alternating graphs. Furthermore, we focus on the SIS model [Het00], which resembles a flu-like virus, where healthy nodes get the virus stochastically from their infected neighbors, and infected nodes get cured with some probability and become susceptible again. The SIS model can be also used in modeling many different types of dynamical processes as well, for example, modeling product penetration in marketing [Rog03].

More specifically, the inputs to our problem are: (a) a set of T alternating graphs, and (b) the infectivity of the virus and the recovery rate (β, δ for the SIS model). We want to

answer the following question (rigorously defined in Section 3.3):

- Q1. Can we say whether a small infection can “take-off” and create an epidemic under the SIS model (i.e. determine the so-called epidemic threshold)?

While epidemic spreading on static graphs has been studied extensively (e.g., see [Het00, AM91, PSV02, CWW⁺08]), virus propagation on time-varying graphs has received little attention. Moreover, most previous studies on time-varying graphs use only simulations [BBE⁺08]. We review in more detail the previous efforts in Section 3.2.

We are arguably the first to study virus propagation *analytically* on arbitrary, and *time-varying* graphs. In more detail, the contributions of our work can be summarized in the following points:

1. We formulate the problem, and show that it can be approximated with a *Non-Linear Dynamical System* (NLDS).
2. We give the **first** closed-formula for the epidemic threshold, involving the first eigenvalue of the so-called *system-matrix* (see Theorem 3.1). The system-matrix combines the connectivity information (the alternating adjacency matrices) and the characteristics of the virus (infectivity and recovery rate).

The rest of the Chapter is organized as follows: We review related work in Section 3.2, explain the formal problem definitions in Section 3.3, and describe the proofs for the threshold and illustrate the theorem in Section 3.4. We discuss the generality of the result in Section 3.7 and finally conclude in Section 3.8.

3.2 Related Work

We have already reviewed the related work concerning epidemic thresholds in the previous chapter. To summarize, none of the earlier related work focus on epidemic thresholds for *arbitrary, real* graphs, with only exceptions of [WCWF03, CWW⁺08], and its follow-up paper by Ganesh et al. [GMT05]. However, even these works [WCWF03, CWW⁺08, GMT05] assume that the underlying graph is fixed, which is unrealistic in many applications. Hence, to the best of our knowledge, including comprehensive epidemiological texts [AM91, Bai75] and well-cited surveys [Het00], we are the **first** to analytically study virus propagation on *arbitrary, real* and *time-varying* graphs.

3.3 Problem Definitions

Table 3.1 lists the main symbols used in this work. Following standard notation, we use capital bold letters for matrices (e.g. \mathbf{A}), lower-case bold letters for vectors (e.g. \mathbf{a}), and calligraphic fonts for sets (e.g. \mathcal{S}) and we denote the transpose with a prime (i.e., \mathbf{A}' is the transpose of \mathbf{A}). In this chapter, we focus on un-directed un-weighted graphs which

Table 3.1: Symbols

Symbol	Definition and Description
$\mathbf{A}, \mathbf{B}, \dots$	matrices (bold upper case)
$\mathbf{A}(i, j)$	element at the i^{th} row and j^{th} column of \mathbf{A}
$\mathbf{A}(i, :)$	i^{th} row of matrix \mathbf{A}
$\mathbf{A}(:, j)$	j^{th} column of matrix \mathbf{A}
\mathbf{I}	standard $n \times n$ identity matrix
$\mathbf{a}, \mathbf{b}, \dots$	column vectors
$\mathcal{J}, \mathcal{J}, \dots$	sets (calligraphic)
n	number of nodes in the graphs
T	number of different alternating behaviors
$\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_T$	T corresponding size $n \times n$ symmetric alternating adjacency matrices
β	virus transmission probability in the SIS model
δ	virus death probability in the SIS model
$\lambda_{\mathbf{M}}$	first eigen-value (in absolute value) of a matrix \mathbf{M}
$\mathbf{u}_{\mathbf{M}}$	corresponding first eigen-vector (for $\lambda_{\mathbf{M}}$) of a matrix \mathbf{M}
$p_{i,t}$	probability that node i is infected at time t
\mathbf{p}_t	$\mathbf{p}_t = (p_{1,t}, p_{2,t}, \dots, p_{n,t})'$
\mathbf{p}_{2t+1}	probability of infection vector for odd days
\mathbf{p}_{2t}	probability of infection vector for even days
η_t	the expected number of infected nodes at time t

we represent by the adjacency matrix and only the SIS virus propagation model. The SIS model is the susceptible/infected/susceptible virus model where β is the probability that an infected node will transmit the infection over a link connected to a neighbor and δ is the probability with which an infected node cures itself and becomes susceptible again. Please see [Het00] for a detailed discussion on SIS and other virus models.

Consider a setting with clearly different behaviors say, day/night, each characterized by a corresponding adjacency matrix (\mathbf{A}_1 for day, \mathbf{A}_2 for night), then what is the epidemic threshold under a SIS virus model? More generally, the problem we are tackling can be formally stated as follows:

Problem 3.1. Epidemic Threshold

Given: (1) T alternating behaviors, characterized by a set of T graphs $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_2 \dots \mathbf{A}_T\}$; and
(2) the SIS model [CWW⁺08] with virus parameters β and δ ;

Find: A condition, under which the infection will die out exponentially quickly (regardless of initial condition).

3.4 Epidemic Threshold on Time-varying Graphs

To simplify discussion, we consider $T = 2$ in Problem 3.1 with \mathcal{A} to consist of only two graphs: G_1 with the adjacency matrix \mathbf{A}_1 for the odd time-stamps (the ‘days’) and G_2 with the adjacency matrix \mathbf{A}_2 for the even time-stamps (the ‘nights’). Our proofs and results can be naturally extended to handle any arbitrary sequence of T graphs.

3.4.1 The NLDS

We first propose to approximate the infection dynamics by a Non-linear dynamical system (NLDS) representing the evolution of the probability of infection vector (\mathbf{p}_t) over time. We can compute the probability $\zeta_t(i)$ that node i does not receive any infections at time t . A node i won’t receive any infection if either any given neighbor is not infected or it is infected but fails to transmit the infection with probability $1 - \beta$. Assuming that the neighbors are *independent*, we get:

$$\begin{aligned}\zeta_{2t+1}(i) &= \prod_{j \in \mathcal{NE}_1(i)} (p_{j,2t+1}(1 - \beta) + (1 - p_{j,2t+1})) \\ &= \prod_{j \in \{1..n\}} (1 - \beta \mathbf{A}_1(i, j) p_{j,2t+1})\end{aligned}\quad (3.1)$$

where $\mathcal{NE}_1(i)$ is the set of neighbors of node i in the graph G_1 with adjacency matrix \mathbf{A}_1 . Similarly, we can write $\zeta_{2t+2}(i)$ as:

$$\begin{aligned}\zeta_{2t}(i) &= \prod_{j \in \mathcal{NE}_2(i)} (p_{j,2t}(1 - \beta) + (1 - p_{j,2t})) \\ &= \prod_{j \in \{1..n\}} (1 - \beta \mathbf{A}_2(i, j) p_{j,2t})\end{aligned}\quad (3.2)$$

So, $p_{i,2t+1}$ and $p_{i,2t+2}$ are:

$$\begin{aligned}1 - p_{i,2t+1} &= \delta p_{i,2t} + (1 - p_{i,2t}) \zeta_{2t}(i) \\ \Rightarrow p_{i,2t+1} &= 1 - \delta p_{i,2t} - (1 - p_{i,2t}) \zeta_{2t}(i)\end{aligned}\quad (3.3)$$

and

$$\begin{aligned}1 - p_{i,2t+2} &= \delta p_{i,2t+1} + (1 - p_{i,2t+1}) \zeta_{2t+1}(i) \\ \Rightarrow p_{i,2t+2} &= 1 - \delta p_{i,2t+1} - (1 - p_{i,2t+1}) \zeta_{2t+1}(i)\end{aligned}\quad (3.4)$$

Note that we can write our NLDS as:

$$\mathbf{p}_{2t+1} = \mathbf{g}_2(\mathbf{p}_{2t}) \quad (3.5)$$

$$\mathbf{p}_{2t+2} = \mathbf{g}_1(\mathbf{p}_{2t+1}) \quad (3.6)$$

where g_1 and g_2 are corresponding non-linear functions as defined by Equations 3.3 and 3.4 (g_1 depends only on \mathbf{A}_1 and g_2 on \mathbf{A}_2).

We have the following theorem about the asymptotic stability of a NLDS at a fixed point:

Theorem 3.1. (Asymptotic Stability, e.g. see [HS74]) *The system given by $\mathbf{p}_{t+1} = g(\mathbf{p}_t)$ is asymptotically stable at an equilibrium point \mathbf{p}^* , if the eigenvalues of the Jacobian $\mathbf{J} = \nabla g(\mathbf{p}^*)$ are less than 1 in absolute value, where,*

$$J_{k,l} = [\nabla g(\mathbf{p}^*)]_{k,l} = \left. \frac{\partial p_{k,t+1}}{\partial p_{l,t}} \right|_{\mathbf{p}_t = \mathbf{p}^*}$$

The fixed point of our interest is the $\mathbf{0}$ vector which is the state when all nodes are susceptible and not infected. We want to then analyze the stability of our NLDS at $\mathbf{p}_{2t} = \mathbf{p}_{2t+1} = \mathbf{0}$. From Equations 3.5 and 3.6, we get:

$$\left. \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t+1}} \right|_{\mathbf{p}_{2t+1}=\mathbf{0}} = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1 = \mathbf{S}_1 \quad (3.7)$$

$$\left. \frac{\partial \mathbf{p}_{2t+1}}{\partial \mathbf{p}_{2t}} \right|_{\mathbf{p}_{2t}=\mathbf{0}} = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2 = \mathbf{S}_2 \quad (3.8)$$

Any eigenvalue $\lambda_{\mathbf{S}_1}^i$ of \mathbf{S}_1 and $\lambda_{\mathbf{S}_2}^i$ of \mathbf{S}_2 ($i = 1, 2, \dots$) is related to the corresponding eigenvalue $\lambda_{\mathbf{A}_1}^i$ of \mathbf{A}_1 and $\lambda_{\mathbf{A}_2}^i$ of \mathbf{A}_2 as:

$$\lambda_{\mathbf{S}_1}^i = (1 - \delta) + \beta\lambda_{\mathbf{A}_1}^i \quad (3.9)$$

$$\lambda_{\mathbf{S}_2}^i = (1 - \delta) + \beta\lambda_{\mathbf{A}_2}^i \quad (3.10)$$

Recall that as \mathbf{A}_1 and \mathbf{A}_2 are symmetric real matrices (the graphs are undirected), from the Perron-Frobenius theorem [McC00], $\lambda_{\mathbf{A}_1}$ and $\lambda_{\mathbf{A}_2}$ are real and positive. So, from Equations 3.9 and 3.10 $\lambda_{\mathbf{S}_1}$ and $\lambda_{\mathbf{S}_2}$ are also real and positive.

3.4.2 The Threshold

We are now in a position to derive the epidemic threshold. First, we have the following lemma:

Lemma 3.1. *If $\lambda_{\mathbf{S}} < 1$, then \mathbf{p}_{2t} dies out exponentially quickly; and $\mathbf{0}$ is asymptotically stable for \mathbf{p}_{2t} , where $\mathbf{S}_1 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1$, $\mathbf{S}_2 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2$ and $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2$.*

Proof. Since $\mathbf{p}_{2t+2} = g_1(g_2(\mathbf{p}_{2t}))$ (from Equations 3.5 and 3.6), we have

$$\begin{aligned} \left. \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t}} \right|_{\mathbf{p}_{2t}=\mathbf{0}} &= \left(\left. \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t+1}} \right|_{\mathbf{p}_{2t+1}=\mathbf{0}} \times \left. \frac{\partial \mathbf{p}_{2t+1}}{\partial \mathbf{p}_{2t}} \right|_{\mathbf{p}_{2t}=\mathbf{0}} \right) \\ &= \left(\left. \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t+1}} \right|_{\mathbf{p}_{2t+1}=\mathbf{0}} \right) \times \left(\left. \frac{\partial \mathbf{p}_{2t+1}}{\partial \mathbf{p}_{2t}} \right|_{\mathbf{p}_{2t}=\mathbf{0}} \right) \\ &= \mathbf{S}_1 \mathbf{S}_2 = \mathbf{S} \end{aligned} \quad (3.11)$$

The first equation is due to chain-rule, second equation is because $\mathbf{p}_{2t} = \mathbf{0}$ implies $\mathbf{p}_{2t+1} = \mathbf{0}$; and the final step is due to Equations 3.7 and 3.8.

Therefore, using Theorem 3.1, we get that if $\lambda_S < 1$, we have that $\mathbf{0}$ is asymptotically stable for \mathbf{p}_{2t} .

We now prove that \mathbf{p}_{2t} in fact goes down exponentially to $\mathbf{0}$ if $\lambda_S < 1$. To see this, after linearizing both g_1 and g_2 at $\mathbf{p}_{2t} = \mathbf{p}_{2t+1} = \mathbf{0}$, we have

$$\begin{aligned} \mathbf{p}_{2t+2} &\leq \mathbf{S}_1 \mathbf{p}_{2t+1} \\ \mathbf{p}_{2t+1} &\leq \mathbf{S}_2 \mathbf{p}_{2t} \end{aligned} \quad (3.12)$$

Doing the above recursively, we have

$$\mathbf{p}_{2t} \leq (\mathbf{S}_1 \mathbf{S}_2)^t \mathbf{p}_0 = (\mathbf{S})^t \mathbf{p}_0 \quad (3.13)$$

Let η_t be the expected number of infected nodes at time t . Then,

$$\begin{aligned} \eta_{2t} = |\mathbf{p}_{2t}|_1 &\leq |(\mathbf{S})^t \mathbf{p}_0|_1 \\ &\leq |(\mathbf{S})^t|_1 |\mathbf{p}_0|_1 = |(\mathbf{S})^t|_1 \eta_0 \\ &\leq \sqrt{n} |(\mathbf{S})^t|_2 \eta_0 = \sqrt{n} \lambda_S^t \eta_0 \end{aligned} \quad (3.14)$$

Therefore, if $\lambda_S < 1$, we have that η_{2t} goes to zero exponentially fast. \square

The above lemma provides the condition for the even time-stamp probability vector to go down exponentially. But, the next lemma shows that this condition is enough to ensure that even the odd time-stamp probability vector to go down exponentially.

Lemma 3.2. *If $\lambda_S < 1$, then \mathbf{p}_{2t+1} dies out exponentially quickly; and $\mathbf{0}$ is asymptotically stable for \mathbf{p}_{2t+1} , where $\mathbf{S}_1 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1$, $\mathbf{S}_2 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2$ and $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2$.*

Proof. Doing the same analysis as in Lemma 3.1, we can see that the condition for \mathbf{p}_{2t+1} to be asymptotically stable and die exponentially quickly is:

$$\lambda_{\mathbf{S}_2 \times \mathbf{S}_1} < 1 \quad (3.15)$$

Now note that as \mathbf{S}_1 and \mathbf{S}_2 are invertible: $\mathbf{S}_1 \times \mathbf{S}_2 = \mathbf{S}_1 \times \mathbf{S}_2 \times \mathbf{S}_1 \times \mathbf{S}_1^{-1}$. But this implies that $\mathbf{S}_2 \times \mathbf{S}_1$ is similar to $\mathbf{S}_1 \times \mathbf{S}_2$ (matrix \mathbf{P} is similar to \mathbf{Q} if $\mathbf{P} = \mathbf{B}\mathbf{Q}\mathbf{B}^{-1}$, for some invertible \mathbf{B}). We know that similar matrices have the same spectrum [GVL89], thus $\mathbf{S}_2 \times \mathbf{S}_1$ and $\mathbf{S}_1 \times \mathbf{S}_2$ have the same eigenvalues. Hence, the condition for exponential die out of \mathbf{p}_{2t+1} and asymptotic stability is the same as that for \mathbf{p}_{2t} which is $\lambda_S < 1$. \square

Lemma 3.1 and Lemma 3.2 imply that this threshold is well-defined in the sense that the probability vector for both the odd and even time-stamps go down exponentially. Thus we can finally conclude the following theorem:

Theorem 3.1. (Epidemic Threshold) *If $\lambda_S < 1$, then \mathbf{p}_{2t} and \mathbf{p}_{2t+1} die out exponentially quickly; and $\mathbf{0}$ is asymptotically stable for both \mathbf{p}_{2t} and \mathbf{p}_{2t+1} , where $\mathbf{S}_1 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1$, $\mathbf{S}_2 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2$ and $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2$. Similarly for any general \mathbb{T} , the condition is:*

$$\boxed{\lambda_{\prod_i \mathbf{S}_i} < 1} \quad (3.16)$$

where $\forall i \in \{1, 2, \dots, \mathbb{T}\} \mathbf{S}_i = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_i$.

We call \mathbf{S} as the *system-matrix* of the system; thus, the first eigenvalue of the system-matrix determines whether a given system is below threshold or not.

3.5 Salient Points

Sanity check: Clearly, when $T = 1$, the system is equivalent to a static graph system with \mathbf{A}_1 and virus parameters β, δ . In this case the threshold is (from Theorem 3.1) $\lambda_{(1-\delta)\mathbf{I}+\beta\mathbf{A}_1} < 1 \Rightarrow \beta\lambda_{\mathbf{A}_1}/\delta < 1$ i.e. we recover the known threshold in the static case [CWW⁺08].

A conservative condition: Notice that from Equations 3.7 and 3.8 and Theorem 3.1, for our NLDS to be fully asymptotically stable at $\mathbf{0}$ (i.e. \mathbf{p}_t decays monotonically), we need the eigenvalues of both \mathbf{S}_1 and \mathbf{S}_2 be less than 1 in absolute value. Hence, $\beta\lambda/\delta < 1$ where $\lambda = \max(\lambda_{\mathbf{A}_1}, \lambda_{\mathbf{A}_2})$ is sufficient for full stability. Intuitively, this argument says that the alternating sequence of graphs can not be worse than static case of having the best-connected graph of the two repeated indefinitely. Let $\lambda_{\mathbf{A}_1} > \lambda_{\mathbf{A}_2}$. Consider a sequence of graphs $\mathcal{S} = \{\mathbf{A}_1, \mathbf{A}_1 \dots\}$ repeating indefinitely instead of our alternating $\{\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_1, \mathbf{A}_2, \dots\}$ sequence. Clearly, if an infection dies exponentially in \mathcal{S} , then it will die exponentially in our original alternating sequence as well because $\lambda_{\mathbf{A}_1} > \lambda_{\mathbf{A}_2}$. The set \mathcal{S} is essentially just the static graph case: hence, if $\beta\lambda_{\mathbf{A}_1}/\delta = \beta\lambda/\delta < 1$, then $\mathbf{0}$ is asymptotically stable for \mathbf{p}_t . The case when $\lambda_{\mathbf{A}_1} < \lambda_{\mathbf{A}_2}$ is similar. But this notion of a threshold is too stringent and *conservative*: it can happen that a stronger virus can still lead to a general exponential decrease instead of a strict monotonous decrease. This is because we forced the eigenvalues of *both* \mathbf{S}_1 and \mathbf{S}_2 to be less than 1 in absolute value here, when we can probably get away with less. Theorem 3.1 precisely formalizes this idea and gives us a more practical condition for a general decreasing trend of every corresponding alternating time-stamp values decaying. We illustrate this further in the experiments.

3.6 Experiments

We will demonstrate our result in this section using simulation experiments. We conducted a series of experiments using the MIT Reality Mining data set [EPL09]. The Reality Mining data consists of 104 mobile devices (cellular phones) monitored over a period of nine months (September 2004 - June 2005). If another participating Bluetooth device was within a range of approximately 5-10 meters, the date and time of the contact and the device's MAC address were recorded. Bluetooth scans were conducted at 5-minute intervals and aggregated into two 12-hour period adjacency matrices (*day* and *night*). The epidemic simulations were accomplished by alternating the *day* and *night* matrices over the period of simulation. All experiments were run on a 64-bit, quad-core (2.5Ghz each) server running a CentOS linux distribution with shared 72 GB of RAM.

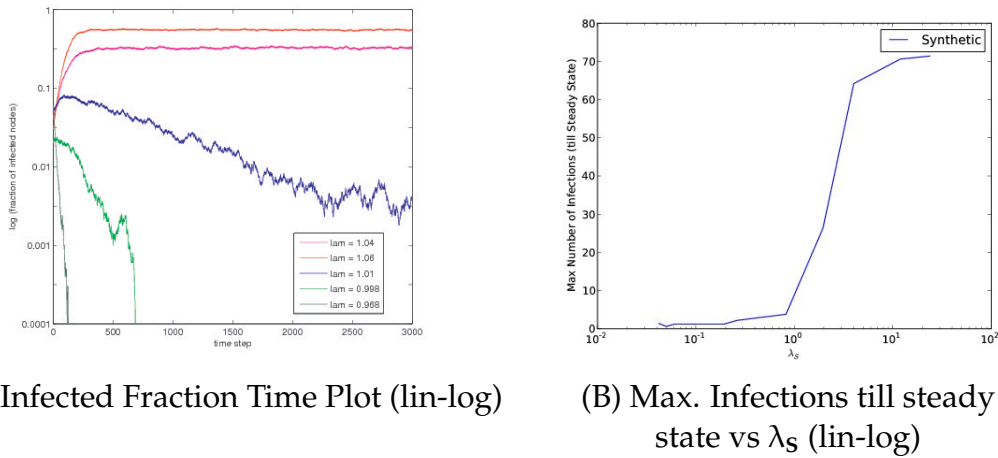


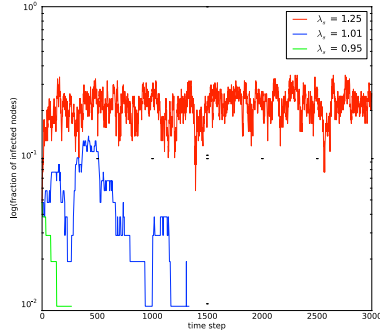
Figure 3.1: SIS simulations on our synthetic example (all values averages over 20 runs) (A) Fraction of nodes infected vs Time-stamp (lin-log scale). Note the qualitative difference in behavior *under* (green) and *above* (red) the threshold. Also, note that the green line is *below* the threshold but is actually *above* the conservative threshold ($\beta\lambda/\delta = 1.100$ here). Hence while both \mathbf{p}_{2t} and \mathbf{p}_{2t+1} decrease exponentially separately, but \mathbf{p}_t itself does not monotonously go down. (B) Plot of Max. number of infected nodes till steady state vs λ_S (by varying β) (lin-log). As predicted by our results, notice that there is a sudden ‘take-off’ and a change of behavior of the curve exactly when $\lambda_S = 1$.

Simulations were conducted using a combination of Matlab 7.9 and Python 2.6.

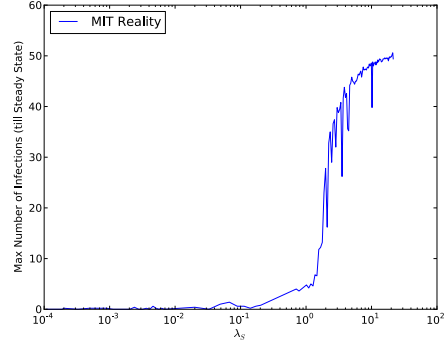
Figures 3.1 and 3.2 demonstrate our result on a synthetic example and graphs from MIT reality data. In the synthetic example, we have 100 nodes, such that G_1 is a full clique (without self loops) whereas G_2 is a chain. All values are average over several runs of the simulations and the infection is started by infecting 5 nodes. In short, as expected from the theorem, the difference in behavior above, below and at threshold can be distinctly seen in the figures.

Figures 3.1(A) and 3.2(A) show the time-plot of number of infections for λ_S values above and below the threshold. While above threshold the infection reaches a steady state way above the starting point, below threshold it decays fast and dies out. In case of Figure 3.1(A), also note the the difference between the conservative threshold and our threshold. The green curve is *below* our threshold but *above* the conservative threshold. But again, as predicted from our theorems, clearly while there are dampening oscillations and the infection decays but \mathbf{p}_t itself does not monotonously go down (and hence the “bumpy” nature of the curve). This exemplifies the practical nature of our threshold and its usefulness as we are more concerned with the general trend of the number of infections curve and not every small ‘bump’ because of the presence of alternating graphs.

Figures 3.1(B) and 3.2(B) show a ‘take-off’ plot showing max. number of infections till steady state (intuitively the ‘footprint’) for different values of λ_S (by varying β). As predicted by our theorem, note the sudden steep change and spike in the size of the footprint when $\lambda_S = 1$ in both the plots.



(A) Infected Fraction Time Plot (lin-log)



(B) Max. Infections till steady state vs λ_S (lin-log)

Figure 3.2: SIS simulations on the MIT reality mining graphs (all values averages over 20 runs) (A) Fraction of nodes infected vs Time-stamp (lin-log scale). Note the qualitative difference in behavior *under* (green) and *above* (red) the threshold. (B) Plot of Max. number of infected nodes till steady state vs λ_S (by varying β) (lin-log). As predicted by our results, notice that there is a sudden ‘take-off’ and a change of behavior of the curve when $\lambda_S = 1$.

3.7 Discussion—Generality of our results

How can we model more complex situations like ‘unequal duration’ behaviors etc.? Note that the alternation period T can be longer than 2 and we can have *repetitions* in the set \mathcal{A} as well e.g., to represent a weekly-style (work day-weekend) alternation we can have $T = 7$ and $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_1, \dots, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_2\}$. Similarly, we can model situations like unequal duration of ‘day’ and ‘night’ i.e. unequal duration of matrices \mathbf{A}_1 and \mathbf{A}_2 . Say, \mathbf{A}_1 is present for only 8 hours at work, while \mathbf{A}_2 is present for the remaining 16 hours at home. Then, thinking of an hour as our time-step i.e. $T = 24$, the set $\mathcal{A} = \{\mathbf{A}_1, \dots, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_2, \dots\}$, where \mathbf{A}_1 occurs 8 times in \mathcal{A} while \mathbf{A}_2 occurs 16 times. All the threshold results carry forward seamlessly in all the above cases.

3.8 Conclusion

In this chapter, we analytically studied virus-spreading (specifically the SIS model) on arbitrary, time-varying graphs. Given a set of T alternating graphs, modeling e.g. the day/night pattern of human behavior, we ask: what is the epidemic threshold? Our main contributions are:

1. We show how to formulate the problem, namely by approximating it with a *Non-Linear Dynamical System* (NLDS).
2. We give the **first** closed-formula for the threshold, involving the first eigenvalue of the *system-matrix* (see Theorem 3.1).

Chapter 4

Competing Viruses: Winner Takes All

We have assumed till now that a single virus is spreading in isolation, without any competition. Instead, in this chapter we look into understanding the spread of multiple competing viruses. Given two competing products (or memes, or viruses etc.) spreading over a given network, can we predict what will happen at the end, that is, which product will 'win', in terms of highest market share? One may naïvely expect that the better product (stronger virus) will just have a larger footprint, proportional to the quality ratio of the products (or strength ratio of the viruses). However, we prove the surprising result that, under realistic conditions, for *any* graph topology, the stronger virus completely wipes-out the weaker one, thus not merely 'winning' but 'taking it all'. In addition to the proofs, we also demonstrate our result with simulations over diverse, real graph topologies, including the social-contact graph of the city of Portland OR (about *31 million* edges and *1 million* nodes) and internet AS router graphs. Finally, we also provide real data about competing products from Google-Insights, like Facebook-Myspace, and we show again that they agree with our analysis.

4.1 Introduction

Given two competing products like iPhone/Android or Blu-ray/HD-DVD, and 'word of mouth' adoption of them, what will happen in the end? This question is of interest in numerous settings. For example, in a biological virus setting, we have the common flu versus avian flu. In a computer virus setting, clever virus authors make sure that their code eliminates most other computer viruses from the victim's disk. The list continues, with competing scientific theories, competing memes ('coke' vs 'soda' vs 'pop'), and many more.

Our main result is that we answer the above question analytically, and we show that 'winner takes all' (WTA), or, more accurately, the weaker product/virus will soon become extinct. The fate of the stronger virus depends on its strength: below the epidemic threshold (more details, later), it will also become extinct, but above that it has good chances of lingering practically for ever. In more detail, we assume

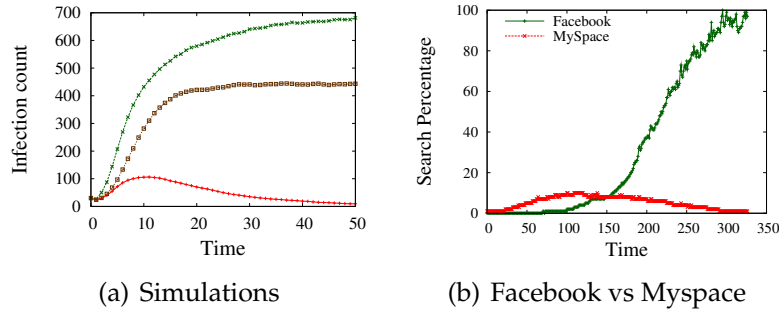


Figure 4.1: (a) Number of infected vs time for simulations on the AS-OREGON network for a virus propagating in isolation (brown square) - note that it is above the epidemic threshold and hence doesn't die out, and the same virus competing with an even stronger virus (green) - note that it now dies-out *completely* (red). (b) Winner takes all in search interest data from Google-Insights for Facebook (green) and Myspace (red). Even though Myspace got a head-start, Facebook wiped it out.

- (a) an SIS-like model (no immunity, like flu),
- (b) perfect mutual immunity (a node can have at most one of the viruses/products, at any given time),
- (c) the underlying network is connected (every node can reach every other node)
- (d) the network is 'fair-play', in the sense that all nodes have the same behavior towards the two competing products/viruses: everybody has the same probability β_1 of getting infected with virus-1 by a sick neighbor, and similarly for virus-2, and for the recovery times.

One of the main contributions is that our theoretical analysis holds for *any* graph topology, while earlier work focuses only on specific-topology graphs (cliques, random, etc).

Figure 4.1(a) gives an illustration of our result: it shows the number of infected nodes vs time for computer simulations on the AS-OREGON network (see Section 4.5 for details) for a 'above-threshold' virus propagating in isolation (brown square) in one case and the virus competing with an even stronger virus (green) in another case. Clearly it is wiped out during the competition, although it gave a fight (red, note the bump). Note that though both the viruses are above the threshold, the weaker virus is wiped out. We prove this result for *arbitrary* underlying networks in this chapter.

Figure 4.1(b) shows the time evolution of search-interest for a pair of competing products Facebook-Myspace. The data came from Google-Insights. Notice that again, the weaker competitor is extinct (or close to that). We will give more case-studies later in Section 4.5.

The outline of the chapter is as follows: we review related work in § 4.2 and formulate the problem giving details of our model in § 4.3. We give the analysis and proof of our WTA result in § 4.4 while we demonstrate it using simulations and real case-studies in § 4.5. Finally, we discuss some subtle issues in § 4.6 and conclude in § 4.7.

4.2 Related Work

We present the related work in this section, which can be categorized into three parts: epidemic thresholds, information diffusion and ecology. We have already summarized prior work in epidemic thresholds for single viruses before. Here we give the rest of the related work. Most of these works either consider only single virus models or typically use only simulation or analyze on very restricted underlying networks.

Information Diffusion Broadly two classes of information cascade models have been proposed (a) independent cascade (IC) [KKT03] (essentially a ‘SIR’ model) and (b) linear threshold (LT) [Gra78]. Research work in multiple cascades has looked into extensions of the IC model with the restriction that nodes can’t switch from one competitor to the other [BKS07, KOW08]. One of the few works to consider switching between the competitors is Pathak et. al. [PBS10]. However, their work differs from ours in several important aspects, as they: (a) use the LT model, as opposed to the ‘flu-like’ SIS model (a cascade style model) we use; (b) assume that nodes may randomly switch between products; (c) do *not* find WTA phenomena; and (d) give no closed-form results - only an algorithm to compute the steady state.

Ecology In ecology, the principle of ‘*competitive exclusion*’ espouses that two species can not occupy the same ecological niche in the long term. Research has gone into studying this using various propagation models like SIS, SIR, Lotka-Volterra [CCHL96, CCHL99, AA05, AM82]. They typically did simulations, or only studied homogenous or structured topologies like cliques.

Distinguishing features of current work: In short, none of the previous work fulfills all the conditions of this current work: (a) analytical proof of ‘WTA’ (b) in arbitrary topologies (c) under a SIS-like model.

4.3 Problem Formulation

In this section, we formulate our problem, giving details about the model used and the assumptions. Table 4.1 explains the terminology we have used in the chapter. Bold letters typically denote matrices (**A**, **C** etc.) or vectors ($\tilde{\mathbf{P}}$, $\tilde{\mathbf{u}}$ etc.).

4.3.1 The propagation model

We assume that the competing viruses are spreading on the network according to a propagation model, which we describe next. We call our propagation model SI_1I_2S , based on the popular “flu-like” SIS (Susceptible-Infected-Susceptible) model [Het00]. SI_1I_2S denotes Susceptible - Infected₁ - Infected₂ - Susceptible. Each node in the graph

Table 4.1: Terms and Symbols

Symbol	Definition and Description
WTA	Winner-Takes-All
SI ₁ I ₂ S	our competing viruses model
β_1 (or β_2)	attack rate of virus 1 (or virus 2)
δ_1 (or δ_2)	cure rate of virus 1 (or virus 2)
\mathbf{A}	adjacency matrix of the underlying graph
$\lambda_{\mathbf{M}}$	set of eigenvalues of the matrix \mathbf{M}
$\lambda_1(\mathbf{M})$	largest eigenvalue of matrix \mathbf{M}
λ	$\lambda_1(\mathbf{A})$
σ_1	$\lambda\beta_1/\delta_1$ (strength of virus 1)
σ_2	$\lambda\beta_2/\delta_2$ (strength of virus 2)
\mathbf{M}^T	transpose of \mathbf{M}
NE(i)	set of neighbors of node i in the graph
\mathbf{I}	identity matrix of appropriate size
$\mathbf{0}$	all-zeros matrix of appropriate size
diag($\vec{\mathbf{P}}$)	the diagonal matrix with elements of vector $\vec{\mathbf{P}}$ in the diagonal

can be in one of three states: Susceptible (healthy), I_1 (infected by virus 1), or I_2 (infected by virus 2). The state transition diagram as seen from a node in the network is shown in Figure 4.2. We could have extended other single virus models as well, but we believe that our model is a reasonable starting point, and we leave the analysis of other models as future work.

Healing (virus death) rate: δ . If a node is in state I_1 (or I_2), it recovers on its own with rate δ_1 (or δ_2). This implies that the time taken for each infected node to heal is exponentially distributed with parameter δ_1 (or δ_2). This parameter captures the persistence of the virus in an inverse way: a high δ means low persistence. For example,

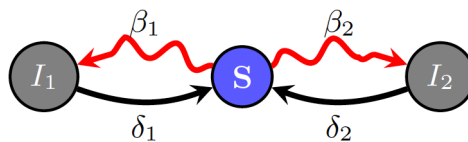


Figure 4.2: State Diagram for a node in the graph under our SI₁I₂S competing viruses model. The node is in the S state if it doesn't have either competitor (say iPhone or Android). It is in I_1 if it gets the iPhone (virus 1) and is in I_2 if it gets the Android (virus 2). The transitions from S to I_1 or I_2 (red-curved arrows) depend on the infected neighbors of the node. The remaining transitions, in contrast, are self-transitions, without the aid of any neighbor.

a very convincing rumor that sticks to one's mind will be modeled with a low δ value.

Attack (virus transmission) rate: β . A healthy node gets infected by infected neighbors, and the virus transmissibility is captured by β_1 and β_2 . Specifically, an infected node transmits its infection to each of its neighbors *independently* at rate β_1 (or β_2). Hence, the time taken for each infected node to transmit the virus to a neighbor over a link is exponentially distributed with parameter β_1 (or β_2).

This is a novel generalization of the single-virus SIS model to a competing-viruses scenario. Note the competition between the viruses: each virus has to compete with the other for healthy victims. Moreover, note that we assume *full mutual immunity*: while a node is infected by one virus, it cannot be infected by the other.

Fair-play: We assume that the competitors are playing a 'fair game': All nodes in the network have the same model parameters (β 's and δ 's) for each of the viruses and behave according to the state-diagram in Figure 4.2.

4.3.2 Problem Statement

We are now in a position to state the problem formally. We assume the underlying network is connected - otherwise we just have separate disconnected problems.

Competing viruses problem

Given: A undirected connected graph G , and the propagation model (SI_1I_2S) parameters (β_1, δ_1 for virus 1, β_2, δ_2 for virus 2)

Find: What will happen at the end i.e. what are the steady-state populations of the two viruses.

4.4 WTA: Results and Proofs

We prove our winner-takes-all (WTA) result on an arbitrary undirected graph in this section. Our main result can be formally stated as follows:

Theorem 4.1 (Winner takes all). *Given an arbitrary undirected, connected graph with adjacency matrix \mathbf{A} and the SI_1I_2S model parameters $(\beta_1, \beta_2, \delta_1, \delta_2)$, then virus 1 will dominate and virus 2 will completely die-out in the steady state if virus 1 is above threshold¹ and the strength of virus 1 is greater than the strength of virus 2 i.e. if $\sigma_1 > 1$ and $\sigma_1 > \sigma_2$.*

The proof is involved, and we present it in the next few pages. We will first prove it for simpler cases of the underlying network - namely a clique and a barbell before we move on to arbitrary graphs.

¹As it follows from Chapter 2, for the *single-virus* SIS model, a virus dies-out unless it is above the epidemic threshold i.e. unless $\beta\lambda/\delta > 1$, where λ is the largest eigenvalue of the adjacency matrix of the underlying graph.

4.4.1 Proof roadmap

In short, the proof has the following steps:

1. **Dynamical System:** construct a suitable dynamical system of differential equations for the propagation process,
2. **Fixed Points:** prove that there are only *three* fixed points and at least one of the viruses has to die out at any fixed point, and
3. **Stability Conditions:** give the precise conditions for each fixed point to be stable (attracting).

Intuitively, the dynamical system generates a field on which we show that only 3 possible fixed points can exist. Moreover the field makes only one of the possible fixed points stable under any given scenario. Figures 4.3(a-c) shows the field-plots in a simple case - when the underlying graph is a clique² of size $N = 1000$. Specifically we show three scenarios (wlog, we assume the first virus is the stronger virus):

BELOW : $1 > \beta_1 N / \delta_1 = 0.6 > \beta_2 N / \delta_2 = 0.2$
(both viruses below the threshold)

MIXED : $\beta_1 N / \delta_1 = 6 > 1 > \beta_2 N / \delta_2 = 0.2$
(one above and one below the threshold)

ABOVE : $\beta_1 N / \delta_1 = 6 > \beta_2 N / \delta_2 = 4 > 1$
(both above the threshold)

The field plots illustrate the fixed points in this setting and their stability. In this case, we have a 2-dimensional field, but for an arbitrary graph it will depend on the number of nodes in the graph. At any point on the field, the direction of the field-arrow tells us where the system will go next. Stable fixed points are marked by bold circles, unstable fixed points by hollow circles, x-axis denotes the # of infected nodes by virus 2 and the y-axis denotes the # infected by virus 1 (the stronger virus). For example, in Figure 4.3(c), both viruses are above threshold, yet the FP_1 and FP_3 points are unstable while the other fixed point corresponding to the stronger virus winning (FP_2) is stable. The trajectory of the simulation is overlaid on the field plots - we can see that the system follows the field lines and is attracted towards and ends up at the stable fixed point in the steady state. We also show the time-evolution separately in Figures 4.3(d-f) - especially note part (f) (ABOVE), virus 2 tries to take over, but is over-powered by virus 1 which goes on to dominate. We can similarly observe the BELOW and MIXED scenarios as well.

We elaborate a bit more on the steps next. Consider a dynamical system (set of differential equations) of the form $x' = F(x)$, where x' is the (component-wise) time derivative of x , and $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is continuous and differentiable. If $F(x_0) = 0$, then x_0 is a stationary point (also called a fixed point). The proof begins by setting up the propagation as a dynamical system of non-linear differential equations and then analyzes the possible fixed points and their stability conditions. In principle, one might expect that

²every node is connected to every other node.

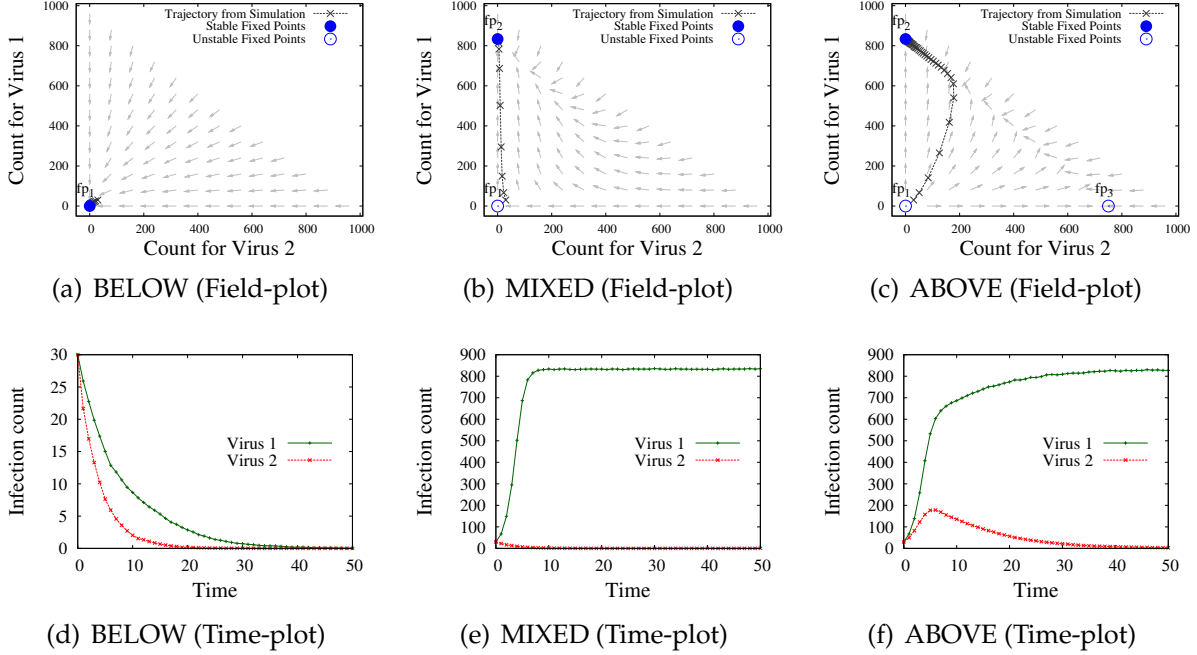


Figure 4.3: (a-c) Field plots for clique of size 1000 for various cases. Stable fixed points are marked by bold circles, unstable fixed points by hollow circles, while the y-axis denotes the number of infected nodes by virus 1 and the x-axis denotes the number infected by virus 2. The simulation trajectory is overlaid. The field plots illustrate the fixed points in this setting and their stability. (d-f) Corresponding Time evolution plots of the competition. Note that virus 2 (red curve) in (f) could have won in isolation, but lost to virus 1.

there might be several fixed points of the system corresponding to different proportions of the populations of the two virus. But we prove that in fact there are always only three fixed points possible and in each at least one virus gets wiped-out.

Further, intuitively, if a fixed point is not stable then the system would be repelled whenever it tries to approach that fixed point. Hence, to fully characterize the fixed points, we need to derive the stability conditions, which give us the conditions for each of these fixed points to be stable and attracting.

For characterizing the stability of the fixed points, we use a well-known result from dynamical system theory (c.f. [HS74]). The fixed points will be a hyperbolic fixed point (i.e. where the linearized stability analysis can be performed) only when none of the eigenvalues of the corresponding Jacobian³ has a zero real part. Further, the system will be stable at a hyperbolic fixed point (attractor) only if the real part of the eigenvalues of the Jacobian is negative. So to make a fixed point a hyperbolic stable attractor we need to impose the condition that *the real parts of all the eigenvalues of the corresponding Jacobian should be negative*. We next show this proof scheme for the special case of a clique

³The Jacobian is the matrix of all component-wise first-order partial derivatives of x' with respect to x evaluated at the fixed point.

topology.

4.4.2 Special case: Clique Topology

In a clique, all nodes are connected to each other with undirected and unweighted edges. Each node is identical to the other and hence our system is a simple continuous time markov chain, due to which we can write down the system equations directly. Let N be the size of the clique and I_1 be the number of nodes infected by virus 1 at some time t . Similarly define I_2 .

Dynamical System: Clearly, under our SI_1I_2S model, we have the following system equations:

$$\begin{aligned}\frac{dI_1}{dt} &= \beta_1(N - I_1 - I_2)I_1 - \delta_1 I_1 \\ \frac{dI_2}{dt} &= \beta_2(N - I_1 - I_2)I_2 - \delta_2 I_2\end{aligned}$$

Fixed Points There are three fixed points of the system of differential equations above (when the rates of change in I_1 and I_2 are zero):

1. $\{I_1 \rightarrow 0, I_2 \rightarrow 0\}$ (i.e. the viruses die-out)
2. $\left\{I_1 \rightarrow N - \frac{\delta_1}{\beta_1}, I_2 \rightarrow 0\right\}$ (i.e. only virus 1 survives)
3. $\left\{I_2 \rightarrow N - \frac{\delta_2}{\beta_2}, I_1 \rightarrow 0\right\}$ (i.e. only virus 2 survives)

Stability Conditions The corresponding Jacobians at the fixed points are:

$$\begin{aligned}1. J_1 &= \begin{bmatrix} N\beta_1 - \delta_1 & 0 \\ 0 & N\beta_2 - \delta_2 \end{bmatrix} \\ 2. J_2 &= \begin{bmatrix} -\delta_1 + \beta_1 \left(N - 2\left(N - \frac{\delta_1}{\beta_1}\right)\right) & -\beta_1 \left(N - \frac{\delta_1}{\beta_1}\right) \\ 0 & \frac{\beta_2 \delta_1}{\beta_1} - \delta_2 \end{bmatrix} \\ 3. J_3 &= \begin{bmatrix} -\delta_1 + \frac{\beta_1 \delta_2}{\beta_2} & 0 \\ -\beta_2 \left(N - \frac{\delta_2}{\beta_2}\right) & -\delta_2 + \beta_2 \left(N - 2\left(N - \frac{\delta_2}{\beta_2}\right)\right) \end{bmatrix}\end{aligned}$$

The eigenvalues of the Jacobians can be seen to be:

$$\begin{aligned}1. \lambda_{J_1} &\equiv \left\{ \beta_1 \left(N - \frac{\delta_1}{\beta_1}\right), \beta_2 \left(N - \frac{\delta_2}{\beta_2}\right) \right\} \\ 2. \lambda_{J_2} &\equiv \left\{ \beta_1 \left(\frac{\delta_1}{\beta_1} - N\right), \beta_2 \left(\frac{\delta_1}{\beta_1} - \frac{\delta_2}{\beta_2}\right) \right\} \\ 3. \lambda_{J_3} &\equiv \left\{ \beta_1 \left(\frac{\delta_2}{\beta_2} - N\right), \beta_2 \left(\frac{\delta_2}{\beta_2} - \frac{\delta_1}{\beta_1}\right) \right\}\end{aligned}$$

From our preceding discussion we know that to have stable fixed points we require that the real part of the eigenvalues of the Jacobians should be negative. Clearly the corresponding conditions for the fixed point to be (a) hyperbolic and (b) stable attractor are:

1. $\frac{\beta_1 N}{\delta_1} < 1$ and $\frac{\beta_2 N}{\delta_2} < 1$
(i.e. both are below threshold)
2. $\frac{\beta_1 N}{\delta_1} > 1$ and $\frac{\beta_1 N}{\delta_1} > \frac{\beta_2 N}{\delta_2}$
(i.e. virus 1 is above threshold and virus 1 strength is greater than virus 2)
3. $\frac{\beta_2 N}{\delta_2} > 1$ and $\frac{\beta_2 N}{\delta_2} > \frac{\beta_1 N}{\delta_1}$
(i.e. virus 2 is above threshold and virus 2 strength is greater than virus 1)

Firstly note that we recover a result similar to the single-virus SIS model case - that if the viruses are below the epidemic threshold, they both die-out. Secondly, we can conclude that in case of a clique, the stronger virus wipes-out the weaker virus if it is above the epidemic threshold.

Non-hyperbolic fixed points: We can see that the fixed points will be non-hyperbolic if the virus strengths are equal. Hence, in this case, no conclusions can be drawn from the linearized analysis and we take a different route. Note that we always have:

$$\int_{I_1^0}^{I_1} \frac{dI_1}{\beta_1 I_1} + \int_0^t \frac{\delta_1}{\beta_1} dt = \int_{I_2^0}^{I_2} \frac{dI_2}{\beta_2 I_2} + \int_0^t \frac{\delta_2}{\beta_2} dt \quad (4.1)$$

$$\Rightarrow \frac{I_1^{\beta_2}}{I_2^{\beta_1}} \times \frac{(I_2^0)^{\beta_1}}{(I_1^0)^{\beta_2}} = e^{\beta_1 \beta_2 (\frac{\delta_2}{\beta_2} - \frac{\delta_1}{\beta_1}) t} \quad (4.2)$$

where I_1^0 and I_2^0 are the initial values of I_1 and I_2 . The R.H.S. will evaluate to one in our case (the virus strengths are equal). Hence now, the ratio of virus populations at any given time t will be directly proportional to the initial ratio (up to some exponents). Also, clearly, the maximal ratios are attained at one of the last two fixed points.

4.4.3 Special Case: Barbell Graph

A barbell graph G has two cliques (say clique C_1 and C_2 of size N each) connected through weak edges. Specifically, we assume that all nodes in C_1 are connected to all nodes in C_2 (and vice versa) with edges of weight ϵ , whereas they are connected with nodes within the same clique with edges of weight 1. In this case, by symmetry we can see that the virus populations in both the cliques should remain the same at the steady state. If we follow the steps in the case of a single clique, we get at steady state:

$$\begin{aligned} \beta_1(N - I_1 - I_2)I_1(1 + \epsilon) &= \delta_1 I_1 \\ \beta_2(N - I_1 - I_2)I_2(1 + \epsilon) &= \delta_2 I_2 \end{aligned}$$

Hence, the only possible fixed points are:

1. $\{I_1 \rightarrow 0, I_2 \rightarrow 0\}$ (i.e. the viruses die-out)
2. $\left\{ I_1 \rightarrow N - \frac{\delta_1}{\beta_1 * (1 + \epsilon)}, I_2 \rightarrow 0 \right\}$ (i.e. only virus 1 survives)
3. $\left\{ I_2 \rightarrow N - \frac{\delta_2}{\beta_2 * (1 + \epsilon)}, I_1 \rightarrow 0 \right\}$ (i.e. only virus 2 survives)

Moreover, continuing similarly as the single clique case, we can see that the stronger virus again wipes-out the weaker virus as long as it is above the epidemic threshold (note that in this case $\lambda = (1 + \epsilon)N$, hence the threshold condition for a single virus is $(1 + \epsilon)\beta N/\delta > 1$).

4.4.4 General Arbitrary Graph

Let \mathbf{A} be the adjacency matrix of the arbitrary graph of N nodes. Let $p_{i,1}$ be the probability of node i to be in the I_1 state. Similarly define $p_{i,2}$ and s_i is the probability of node i being in the susceptible state. Clearly, $s_i + p_{i,1} + p_{i,2} = 1$.

Dynamical System: As we have a continuous time process, the following system equations hold for each node i :

$$\begin{aligned}\frac{dp_{i,1}}{dt} &= -\delta_1 p_{i,1} + \beta_1(1 - p_{i,1} - p_{i,2}) \sum_j (\mathbf{A}_{ij} \mathbf{1}_{j,1}) \\ \frac{dp_{i,2}}{dt} &= -\delta_2 p_{i,2} + \beta_2(1 - p_{i,1} - p_{i,2}) \sum_j (\mathbf{A}_{ij} \mathbf{1}_{j,2})\end{aligned}$$

where $\mathbf{1}_{j,k}$ (for $k = 1, 2$) is the indicator random variable denoting if node j is infected with virus k . Our system is *not* a markov chain due to the presence of random variables $\mathbf{1}_{j,k}$ in the rate equations. But after making a mean-field approximation ($\mathbf{1}_{j,1} \approx E[\mathbf{1}_{j,1}] = p_{j,1}$ and $\mathbf{1}_{j,2} \approx p_{j,2}$, where $E[X]$ is the expected value of the random variable X), we get the following dynamical system:

$$\frac{dp_{i,1}}{dt} = -\delta_1 p_{i,1} + \beta_1(1 - p_{i,1} - p_{i,2}) \sum_j (\mathbf{A}_{ij} p_{j,1}) \quad (4.3)$$

$$\frac{dp_{i,2}}{dt} = -\delta_2 p_{i,2} + \beta_2(1 - p_{i,1} - p_{i,2}) \sum_j (\mathbf{A}_{ij} p_{j,2}) \quad (4.4)$$

(for each node i).

Fixed Points: At the steady state i.e. at fixed points where the change in probabilities will be zero, we get (for each node i):

$$\delta_1 p_{i,1} = \beta_1(1 - p_{i,1} - p_{i,2}) \sum_j (\mathbf{A}_{ij} p_{j,1}) \quad (4.5)$$

$$\delta_2 p_{i,2} = \beta_2(1 - p_{i,1} - p_{i,2}) \sum_j (\mathbf{A}_{ij} p_{j,2}) \quad (4.6)$$

which can be written in vector-form as:

$$\beta_1 \mathbf{SA} \tilde{\mathbf{P}}_1 = \delta_1 \tilde{\mathbf{P}}_1 \quad (4.7)$$

$$\beta_2 \mathbf{SA} \tilde{\mathbf{P}}_2 = \delta_2 \tilde{\mathbf{P}}_2 \quad (4.8)$$

where $\tilde{\mathbf{P}}_1 = [p_{1,1}, p_{2,1}, \dots, p_{N,1}]^\top$, $\tilde{\mathbf{P}}_2 = [p_{1,2}, p_{2,2}, \dots, p_{N,2}]^\top$ and $\mathbf{S} = \text{diag}(s_i) = \mathbf{I} - \text{diag}(\tilde{\mathbf{P}}_1 + \tilde{\mathbf{P}}_2)$.

In all of the following analysis, we assume we are operating at fixed point unless stated otherwise, i.e. Equations 4.5 and 4.6 or equivalently Equations 4.7 and 4.8 hold. Additionally, we assume that \mathbf{A} is connected. First we have the following series of lemmas.

Lemma 4.1. $\forall i$ we have that $s_i \neq 0$.

Proof. If $s_i = 0$ for any i , then Equations 4.5 and 4.6 immediately imply that $p_{i,1} = p_{i,2} = 0$ which contradicts $s_i + p_{i,1} + p_{i,2} = 1$. \square

Lemma 4.2. *If $\exists i$ $p_{i,1} = 0 \Rightarrow \forall i$ $p_{i,1} = 0$. Similarly $\exists i$ $p_{i,2} = 0 \Rightarrow \forall i$ $p_{i,2} = 0$.*

Proof. If $\exists i$ $p_{i,1} = 0$, then from Equation 4.5 we have $\sum_j (\mathbf{A}_{ij} p_{j,1}) = 0$ (as $s_i \neq 0$ from Lemma 4.1). Clearly, \mathbf{A}_{ij} 's are positive only for those nodes j which are neighbors of node i , i.e. for $j \in \text{NE}(i)$ (and there is at least one such j as the graph is connected). For these j , as $p_{j,1}$ can not be negative (they are probabilities), they have to be zero so that the above is true. Now we can apply the same argument we applied for node i in turn for all the neighbors $j \in \text{NE}(i)$ and so on. Finally we get that $\forall i$ $p_{i,1} = 0$ as the graph is connected. We can prove similarly for $p_{i,2}$. \square

Lemma 4.3. *The matrix \mathbf{SA} is non-negative and irreducible.*

Proof. \mathbf{A} is symmetric and irreducible as it is connected. From Lemma 4.1 we have that \mathbf{S} is a diagonal positive matrix. Clearly, it follows that $\mathbf{S} \cdot \mathbf{A}$ maybe asymmetric but it is a non-negative and irreducible matrix (intuitively, multiplying by \mathbf{S} preserves the original edges in \mathbf{A}). \square

Lemma 4.4. *The matrix \mathbf{SA} has a unique positive real number (say λ) as its largest eigenvalue (in magnitude). Further the algebraic multiplicity of λ is 1 and it has a positive eigenvector (say $\tilde{\mathbf{v}}$: then all components of $\tilde{\mathbf{v}}$ are positive).*

Proof. As \mathbf{SA} is non-negative and irreducible (Lemma 4.3), we can apply the Perron-Frobenius theorem [McC00]. The lemma follows directly then. \square

Lemma 4.5. *There are no positive eigenvectors of \mathbf{SA} other than $\tilde{\mathbf{v}}$ (the Perron eigenvector of \mathbf{SA} corresponding to the largest eigenvalue).*

Proof. From Lemma 4.3, it follows that $(\mathbf{SA})^\top$ is non-negative and irreducible as well. Moreover, note that the eigenvalues of any matrix \mathbf{M} and \mathbf{M}^\top are the same. Hence, again applying the Perron-Frobenius theorem to $(\mathbf{SA})^\top$, we have the largest eigenvalue as λ and the corresponding positive eigenvector as say $\tilde{\mathbf{u}}$. From the eigenvalue equation, we know that $\tilde{\mathbf{u}}^\top \mathbf{SA} = \lambda \tilde{\mathbf{u}}^\top$.

Now suppose we have another positive eigenvector, say $\tilde{\mathbf{w}}$, corresponding to eigenvalue t of \mathbf{SA} (so, $\mathbf{SA}\tilde{\mathbf{w}} = t\tilde{\mathbf{w}}$). Then:

$$\lambda\tilde{\mathbf{u}}^T\tilde{\mathbf{w}} = \tilde{\mathbf{u}}^T\mathbf{SA}\tilde{\mathbf{w}} = t\tilde{\mathbf{u}}^T\tilde{\mathbf{w}}$$

Hence $(\lambda - t)\tilde{\mathbf{u}}^T\tilde{\mathbf{w}} = 0$. But $\tilde{\mathbf{u}}^T\tilde{\mathbf{w}} \neq 0$ as both $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{w}}$ are positive. Hence $t = \lambda$. But λ has multiplicity 1 (Lemma 4.4) and hence $\tilde{\mathbf{w}} = \tilde{\mathbf{v}}$. \square

Lemma 4.6. *At fixed point, $\tilde{\mathbf{P}}_1 > 0$ and $\tilde{\mathbf{P}}_2 > 0$ both can not hold unless $\frac{\delta_1}{\beta_1} = \frac{\delta_2}{\beta_2}$.*

Proof. Together with Lemma 4.2, Equation 4.7 implies that either $\tilde{\mathbf{P}}_1 = 0$ or it is a positive eigenvector of \mathbf{SA} with eigenvalue δ_1/β_1 . Similarly from Equation 4.8 (and Lemma 4.2) we get that either $\tilde{\mathbf{P}}_2 = 0$ or it is a positive eigenvector of \mathbf{SA} with eigenvalue δ_2/β_2 . From Lemma 4.5, the only positive eigenvector of \mathbf{SA} is the one corresponding to the largest eigenvalue. Hence both $\tilde{\mathbf{P}}_1 > 0$ and $\tilde{\mathbf{P}}_2 > 0$ can hold only if $\frac{\delta_1}{\beta_1} = \frac{\delta_2}{\beta_2}$. Otherwise at least one of them is zero. \square

Assuming the virus strengths are not equal, Lemma 4.6 implies the following theorem:

Theorem 4.2. *Assuming the virus strengths are not equal, the system has only the following possible fixed points:*

1. $\{\tilde{\mathbf{P}}_1 \rightarrow 0, \tilde{\mathbf{P}}_2 \rightarrow 0\}$ (i.e. the viruses die-out)
2. $\{\tilde{\mathbf{P}}_1 \rightarrow \text{perron eigenvector of } \mathbf{SA}, \tilde{\mathbf{P}}_2 \rightarrow 0\}$ (i.e. only virus 1 survives)
3. $\{\tilde{\mathbf{P}}_2 \rightarrow \text{perron eigenvector of } \mathbf{SA}, \tilde{\mathbf{P}}_1 \rightarrow 0\}$ (i.e. only virus 2 survives)

We can assert the next lemma immediately:

Lemma 4.7. *The second and third fixed points in Theorem 4.2 require $\sigma_1 > 1$ and $\sigma_2 > 1$ respectively.*

Proof. In the second fixed point, virus 2 dies-out and only virus 1 survives. Hence the system now is equivalent to a single virus operating on the whole graph under the standard flu-like SIS model. For this we already know that the virus should be above the ‘epidemic threshold’ if it has to survive (and not die-out exponentially quickly) [CWW⁺08, PCF⁺11]. Hence $\lambda\beta_1/\delta_1 = \sigma_1 > 1$ is necessary for the second fixed point. Similarly we can prove the case for when virus 2 survives. \square

Stability Conditions: We first compute the Jacobian at each of the fixed points.

Lemma 4.8. *The Jacobians at the three fixed points can be written as below. (each Jacobian is a $2N \times 2N$ matrix, each sub-matrix block below is a matrix of size $N \times N$).*

1. $\mathbf{J}_1 = \begin{bmatrix} \beta_1\mathbf{A} - \delta_1\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \beta_2\mathbf{A} - \delta_2\mathbf{I} \end{bmatrix}$
2. $\mathbf{J}_2 = \begin{bmatrix} \beta_1\mathbf{SA} - \delta_1\mathbf{I} - \beta_1\text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1) & -\beta_1\text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1) \\ \mathbf{0} & \beta_2\mathbf{SA} - \delta_2\mathbf{I} \end{bmatrix}$

$$3. \mathbf{J}_3 = \begin{bmatrix} \beta_1 \mathbf{SA} - \delta_1 \mathbf{I} & \mathbf{0} \\ -\beta_2 \text{diag}(\mathbf{A}\tilde{\mathbf{P}}_2) & \beta_2 \mathbf{SA} - \delta_2 \mathbf{I} - \beta_2 \text{diag}(\mathbf{A}\tilde{\mathbf{P}}_2) \end{bmatrix}$$

Here, in \mathbf{J}_2 , $\tilde{\mathbf{P}}_1$ is the Perron eigenvector of \mathbf{SA} with eigenvalue δ_1/β_1 (i.e. it satisfies Equation 4.7 and is non-zero). Similarly $\tilde{\mathbf{P}}_2$ in \mathbf{J}_3 .

Proof. Can be computed using standard differentiation. Details omitted for brevity. \square

Given the discussion before, we can analyze the corresponding conditions for the fixed point to be hyperbolic stable attractor.

Lemma 4.9. *The conditions for the fixed points to be hyperbolic and stable attractor are:*

1. $\sigma_1 < 1$ and $\sigma_1 < 1$
2. $\sigma_1 > \sigma_2$
3. $\sigma_2 > \sigma_1$

Proof. We prove the conditions for each fixed point separately below (we omit some details for brevity):

1. The eigenvalues of matrix \mathbf{J}_1 are simply the eigenvalues of the matrices $\mathbf{M}_1 = \beta_1 \mathbf{A} - \delta_1 \mathbf{I}$ and $\mathbf{M}_2 = \beta_2 \mathbf{A} - \delta_2 \mathbf{I}$. The real part of all the eigenvalues of these matrices will be negative if the real part of the largest eigenvalue is negative (as \mathbf{M}_1 and \mathbf{M}_2 are real and symmetric, all their eigenvalues are real). Hence the conditions for this are $\beta_1 \lambda / \delta_1 < 1$ and $\beta_2 \lambda / \delta_2 < 1$, where λ is the largest eigenvalue of \mathbf{A} .
2. We can see that the eigenvalues of the matrix \mathbf{J}_2 are either the eigenvalues of matrix $\mathbf{M}_1 = \beta_1 \mathbf{SA} - \delta_1 \mathbf{I} - \beta_1 \text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1)$ or the eigenvalues of the matrix $\mathbf{M}_2 = \beta_2 \mathbf{SA} - \delta_2 \mathbf{I}$. The eigenvalues of \mathbf{M}_2 are just the eigenvalues of $\beta_2 \mathbf{SA}$ subtracted by δ_2 . From Lemma 4.4 and Equation 4.7 we know that under this fixed point, the largest eigenvalue of \mathbf{SA} is δ_1/β_1 . This implies that $\Re(\lambda_{\mathbf{SA}}) < \delta_1/\beta_1$ for any eigenvalue $\lambda_{\mathbf{SA}}$ of \mathbf{SA} ⁴. Thus,

$$\Re(\lambda_{\mathbf{M}_2}) = \beta_2 \Re(\lambda_{\mathbf{SA}}) - \delta_2 < \beta_2 \delta_1 / \beta_1 - \delta_2 < 0$$

where the last step follows if $\beta_2/\delta_2 < \beta_1/\delta_1$. Hence, if $\sigma_2 < \sigma_1$, the real part of all the eigenvalues of \mathbf{M}_2 are negative.

Consider the matrix $\mathbf{D} = \mathbf{M}_1 + \mathbf{M}_1^\top = \beta_1(\mathbf{SA} + \mathbf{AS}) - 2\delta_1 \mathbf{I} - 2\beta_1 \text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1)$. Matrix \mathbf{D} is clearly real and symmetric and so has all real eigenvalues. Due to Lemma 4.4 we can apply the Perron-Frobenius theorem to $\mathbf{SA} + \mathbf{AS}$ as well and deduce that its largest eigenvalue $\lambda_1(\mathbf{SA} + \mathbf{AS})$ is positive. Further, from matrix theory [HJ91, HW97], we know that for any real non-negative matrix \mathbf{C} , $\lambda_1(\mathbf{C} + \mathbf{C}^\top) \leq 2\lambda_1(\mathbf{C})$. Hence $0 < \lambda_1(\mathbf{SA} + \mathbf{AS}) \leq 2\lambda_1(\mathbf{SA}) = 2\delta_1/\beta_1$. Again we know from standard linear

⁴ $\Re(x)$ denotes the real part of x

algebra [HJ91], that $\lambda_1(\mathbf{X} + \mathbf{Y}) \leq \lambda_1(\mathbf{X}) + \lambda_1(\mathbf{Y})$ if \mathbf{X} and \mathbf{Y} are symmetric. Hence,

$$\begin{aligned}\lambda_1(\mathbf{D}) &\leq \beta_1\lambda_1(\mathbf{SA} + \mathbf{AS}) - 2\delta_1 - 2\beta_1\lambda_1(\text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1)) \\ &\leq 2\beta_1\delta_1/\beta_1 - 2\delta_1 - 2\beta_1\lambda_1(\text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1)) \\ &\leq -2\beta_1\lambda_1(\text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1)) \\ &< 0\end{aligned}$$

as under this fixed point, $\text{diag}(\mathbf{A}\tilde{\mathbf{P}}_1)$ is a diagonal matrix with positive entries and hence has all positive eigenvalues. As \mathbf{D} has all real eigenvalues, $\lambda_1(\mathbf{D}) < 0$ implies that it has all negative eigenvalues. The Lyapunov theorem [HS74] states that a matrix \mathbf{C} is stable (has $\mathbb{R}(\lambda_{\mathbf{C}}) < 0$) if $\mathbf{C}^T + \mathbf{C}$ has all negative eigenvalues. Applying it to our case, we can see that matrix \mathbf{D} having all negative eigenvalues implies that \mathbf{M}_1 is stable unconditionally under this fixed point.

Finally, as \mathbf{M}_1 and \mathbf{M}_2 both (and so \mathbf{J}_2 as well) have the real part of their eigenvalues negative under the condition $\sigma_2 < \sigma_1$, the fixed point is a hyperbolic stable attractor if $\sigma_2 < \sigma_1$.

3. Analogous to the case of the fixed point above.

Proved. □

Lemma 4.9 combined with Lemma 4.7 allows us to conclude the following:

Theorem 4.3. *The corresponding conditions for each of the fixed points to (a) exist, and (b) have stability (i.e. be a hyperbolic and stable attractor) are:*

1. $\sigma_1 < 1$ and $\sigma_2 < 1$
(i.e. both are below threshold)
2. $\sigma_1 > 1$ and $\sigma_1 > \sigma_2$
(i.e. virus 1 is above threshold and virus 1 strength is greater than virus 2)
3. $\sigma_2 > 1$ and $\sigma_2 > \sigma_1$
(i.e. virus 2 is above threshold and virus 2 strength is greater than virus 1)

Combining Theorem 4.2 and Theorem 4.3, we again have a result similar to the single virus epidemic threshold - that viruses die-out if they are below the individual epidemic threshold (i.e. if $\beta\lambda/\delta < 1$). Finally, they also imply our WTA result (Theorem 4.1).

4.5 Experiments

We demonstrate our result using (a) simulation experiments on varied datasets; and (b) case studies using real data in this section.

4.5.1 Setup

We first briefly describe our experimental setup for the simulations as well as the case studies.

Simulations: WLOG, in our experiments, we assumed that the first virus is the stronger virus. We then considered the following three cases:

BELOW : $1 > \beta_1\lambda/\delta_1 = 0.6 > \beta_2\lambda/\delta_2 = 0.2$

(both viruses below the threshold)

MIXED : $\beta_1\lambda/\delta_1 = 6 > 1 > \beta_2\lambda/\delta_2 = 0.2$

(one above and one below the threshold)

ABOVE : $\beta_1\lambda/\delta_1 = 6 > \beta_2\lambda/\delta_2 = 4 > 1$

(both above the threshold)

We used the following real-world and synthetic network datasets for the simulations:

1. AS-OREGON: The Oregon AS router graph which is a network graph collected from the Oregon router views. It contains 15,420 links among 3,995 AS peers. More information can be found from <http://topology.eecs.umich.edu/data.html>.
2. PORTLAND: One of the biggest available physical contact graphs, representing a synthetic population of the city of Portland, Oregon, USA [NDS07], and has been used in smallpox modeling studies [EGAK⁺04]. It is a social-contact graph containing more than 31 million links (interactions) among about 1.6 million nodes (people).
3. Clique: A fully connected clique of 1000 nodes.
4. Barbell: Two cliques of 500 nodes joined together by weak edges of weight $\epsilon = 0.01$ (see Section 4.4.3 for a description).

We implemented our competing viruses model SI_1I_2S as an event based discrete simulation in C++. We randomly infect 30 nodes for each of the viruses at the start of any simulation. All simulations were run over 1000 time steps and the plots show averaged results from 100 runs.

Case-studies: We collected historical data for 'web-search interest' for various competing products from the Google-Insights website⁵ which aims to 'provide insights into broad search patterns'. This allows us to use the data as a proxy for product sales/adoption for each product. We used the following pairs of rival products:

1. Reddit and Digg: Two social news websites, where users post links to interesting memes/news articles.⁶

⁵www.google.com/insights/search/

⁶www.reddit.com, www.digg.com

2. Facebook and Myspace: Two social network websites, where users add their friends and share posts, pictures etc.⁷
3. Blu-ray and HD-DVD: Two rival competing standards of high-density optical media.

The full mutual immunity model doesn't describe all the above situations perfectly, but it is a very good approximation. We understand that not all of the pairs are mutually exclusive in the strict sense e.g., people can go and put links on both Digg and Reddit, however, people are unlikely to be part of both communities as they have to choose a site while sharing content.

4.5.2 Simulation Results

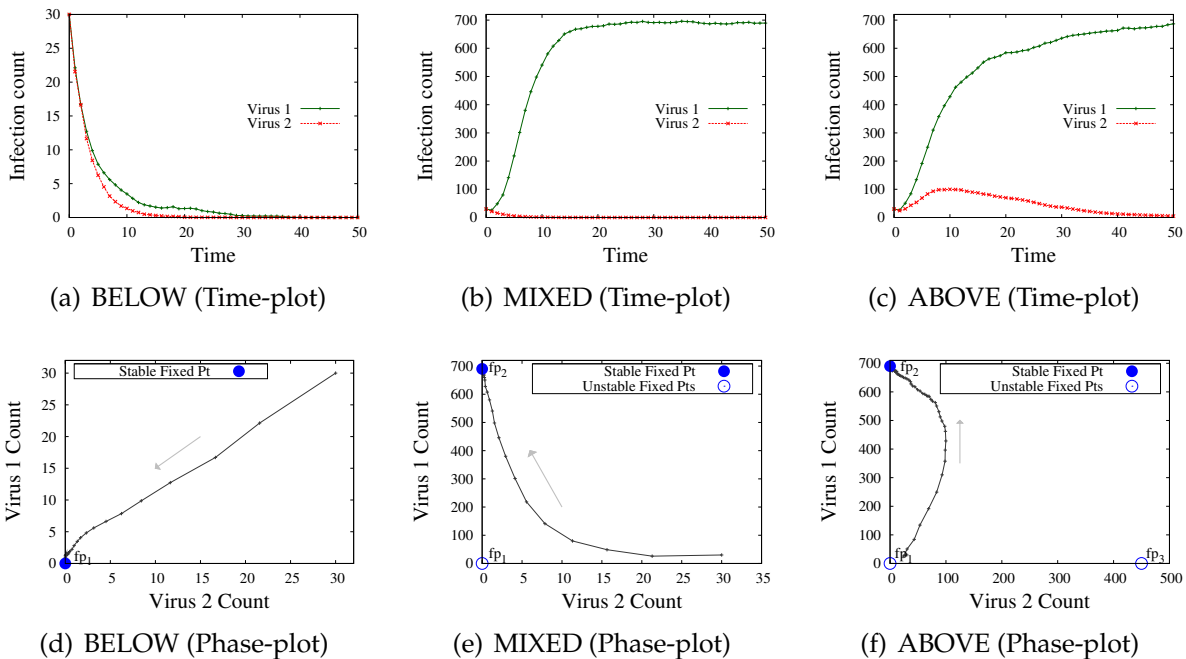


Figure 4.4: (a-c) Number of infected vs time plots for simulations on the AS-OREGON network for different scenarios. (d-f) Corresponding Phase plots (scatter plot of number of infected nodes by virus 1 (y-axis) and number of infected nodes by virus 2). Stable fixed points are marked by bold circles, unstable by hollow circles. Clearly, the stronger virus wins (as long as it is above threshold) and the weaker dies-out completely as our result predicted.

Figures 4.4 and 4.5 demonstrate our results. In short, the plots agree exactly with our result, as expected.

Figure 4.4 shows the *Time-plots* and *Phase-plots* for simulations on the AS-OREGON graph for our three scenarios as discussed before in the setup. The time-plots show the

⁷www.facebook.com, www.myspace.com

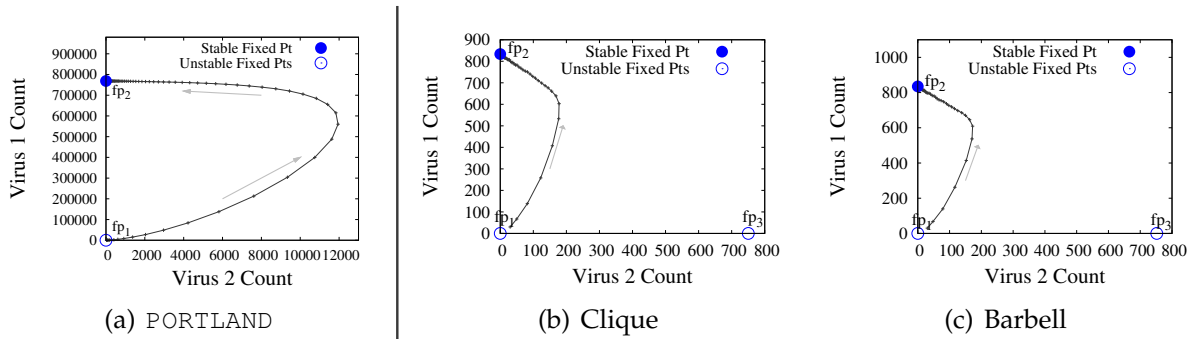


Figure 4.5: Phase plots (scatter plot of number of infected nodes by virus 2 and number of infected nodes by virus 1) plots for simulations on (a) the `PORTLAND` network, (b) a clique and (c) a barbell graph for scenario ABOVE. Again, stable fixed points are marked by bold circles and unstable fixed points by hollow circles (FP₃ not shown in (a) for sake of clarity of the trajectory). The weaker virus tries to dominate (note the bulge), but it dies-out completely and the stronger virus wins, as our result predicted.

Number of nodes Infected vs Time for each of the viruses (red for the weaker virus, green for the stronger one). The phase plot is the scatter plot of number of infected nodes by the stronger virus on the y-axis and number of infected nodes by the weaker virus on the x-axis. Thus a phase plot shows the trajectory of the simulation in the 2-d plane. The stable points in each scenario are marked with solid circles.

In the BELOW case, we expect that both of them die-out. This is borne out by both the time and phase plots (Figures 4.4(a) and (d)). Point FP₁(0, 0) is the only stable fixed point in this case and hence the system converges to it very quickly (see the phase plot). On the other hand, when the stronger virus is above threshold (MIXED) we can see that it takes-over and the other virus dies-out (Figures 4.4(b) and (e)). In this case, point FP₂ is stable and attracting while FP₁ becomes unstable. As a result, we converge to the steady state where only the stronger survives. Finally, in case ABOVE, when each could have dominated in isolation, the stronger virus clearly wins and wipes-out the weaker virus (Figures 4.4(c) and (f)). Here, FP₂ is again stable while the other fixed points are unstable. Moreover, note that the stronger virus reaches the same steady-state as in MIXED. This agrees with our analysis as well (see Lemma 4.7): in both scenarios, the stronger virus will reach the same fixed-point as it would have if operating in isolation, without the presence of a competitor.

Similarly, Figure 4.5 shows the phase-plots for simulations on the other graph datasets - `PORTLAND`, `Clique` and `Barbell`. For lack of space, we just show the plots for case ABOVE. As before, the stronger virus wins and the weaker virus dies-out completely, no matter the network, in perfect agreement with our result (Theorem 4.1).

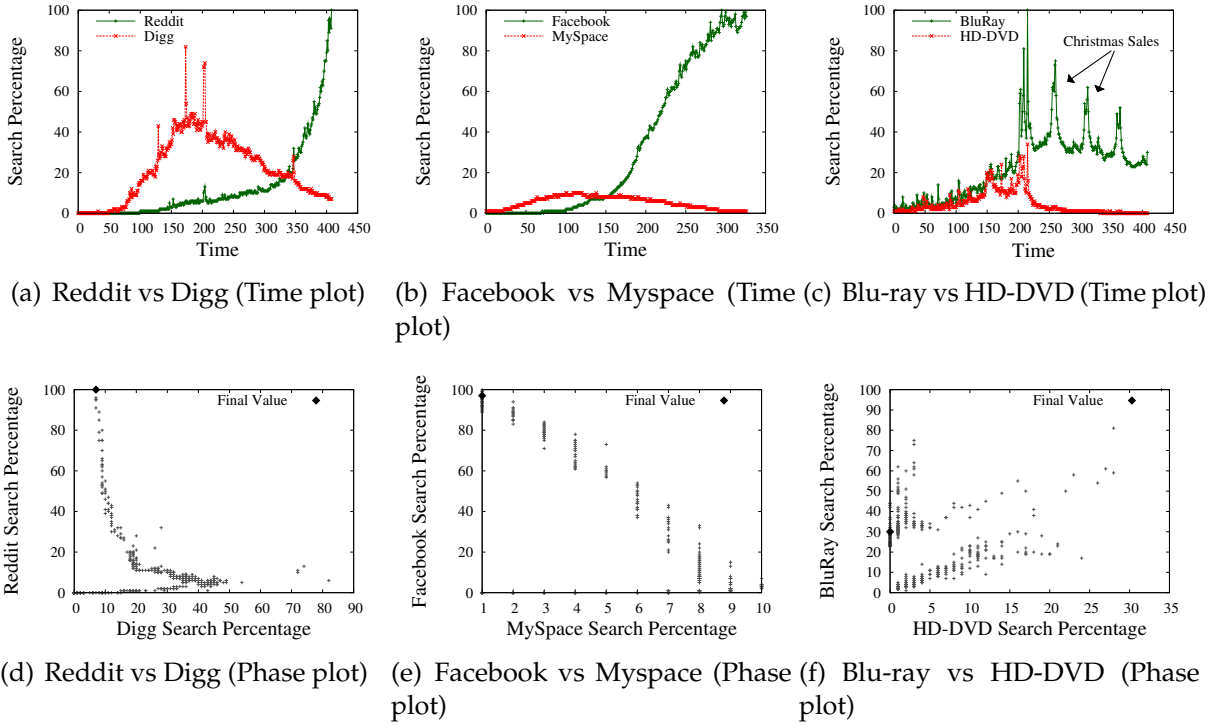


Figure 4.6: (a-c) Real web-search interest vs time plots for pair of competitors (see Section 4.5.1 for more details) (d-f) Corresponding Phase plots. As expected from our WTA result, note that the stronger rival dominates and weaker product almost dies-out.

4.5.3 Case-Studies using Real Data

Figure 4.6 shows the historical data we collected from Google-Insights. In short, they provide corroboration to the WTA phenomenon in real-world as well.

Figures 4.6(a-c) show the web-search interest vs time for the three pairs of competitors we discussed before in the setup. Figures 4.6(d-f) show the corresponding phase-plots (the final data-point is marked by a diamond). Firstly, as it is real data, due to various reasons they do show significant deviations over the smooth steady states observed from our models (e.g., the spikes in Figure 4.6(c) denote Christmas shopping sales). Nevertheless, they broadly give positive evidence for the WTA result e.g., in (a-b) and (d-e), even though Digg and MySpace had a head-start and even dominate for a while, the stronger product (Reddit and Facebook) eventually takes-over. The phase plots also show the trajectories in effect similar to the ones found in our simulations. Clearly, in all the plots we can see that the eventual winner and dominant competitor (Reddit, Facebook, Blu-ray) almost completely wipes-out the weaker competitor, just as our result predicts.

4.6 Discussion

There are several subtle points, that we deferred until now, for clarity of exposition. Specifically, here we discuss the following issues:

Question: Explain the counter-examples, of ‘winner takes all’. If ‘winner takes all’, how come there are competing products where the weaker one still has a non-trivial market share, like ‘Windows’, ‘Mac-os’ (and ‘Linux’)?

Answer: Not ‘level-fields’; or not enough time. There are indeed numerous cases where two (or more) competing products or ideas, co-exist. For example, in the OS ‘wars’, MS-windows has a large market share, with mac-OS having a smaller, but near-constant market share. There are several settings that could cause such deviations.

- One is that we are violating our assumption of ‘fair-play’, e.g., some nodes (like enthusiastic AppleTM fans) exhibit much lower infection probability β , or even zero for one of the viruses. Thus by catering to just that niche where it is much stronger, the competitor can survive.
- A second cause is weak connectivity, like a bar-bell graph with a narrow bridge, and not enough time to reach steady-state.
- A third cause is viruses of near-equal strength. We omit the simulation results here, but similar-strength viruses take too long to reach WTA. This is analogous to the case of two near-equal-strength tennis players, that need several games, and several tie-breakers, before a winner emerges.

Question: Has this WTA phenomenon appeared in other settings?

Answer: Yes, with simulation results. In epidemiology studies, WTA is referred to as ‘competitive exclusion’ e.g. see [CCHL96, CCHL99, AA05, AM82]. However, they typically did simulations, or they only studied homogenous or very structured topologies like cliques.

Question: How about other propagation models (SIR etc)? Will WTA, then?

Answer: We conjecture that the answer is ‘yes’. The full analysis for SIR (= life-long immunity, like mumps) SIRS (= long, but not permanent, immunity) and more, are the focus of our ongoing research. We conjecture that similar results may hold, too, extrapolating from the results of (Prakash et al. [PCF⁺11]): that work showed that, for a single virus, the epidemic threshold has the same form, for almost any virus propagation model.

Question: Will WTA hold, under *partial* mutual immunity?

Answer: Future work - no conjectures. In this work, we assume full mutual exclusion, that is a given node will have at most one of the two viruses/products (iPhone/Android), at any given point in time, but not both. There are marketing, and biological settings that a person may have both products/viruses. Will WTA hold, then? This seems like a difficult question, and left for future work. We suspect that the answer will not be a simple ‘yes’ or ‘no’.

4.7 Conclusions

In summary, we tackled the setting of two competing products (or viruses or ideas etc.) spreading over a network and studied the problem of what happens in the end (i.e. in the steady state). In addition to problem formulation and getting ecological concepts to web-like phenomena, the main contributions of our work are as follows:

1. *WTA Result and Proof*: We provided a theoretical analysis of the propagation model for *arbitrary* graph topology, proving that the 'winner-takes-all' i.e. the stronger virus dominates and wipes-out the weaker virus (if it is above threshold). See Theorem 4.1.
2. *Experiments and Case-studies*: We also demonstrated our result using extensive simulations on real and synthetic networks showing that they match exactly with our predictions. Moreover, using case-studies of historical data of competing products (Blu-ray/HD-DVD, Facebook/MySpace, Reddit/Digg), we provided positive evidence of WTA in real-life.

Chapter 5

Competing Viruses: Co-existence

In the previous chapter, we studied competing viruses spreading over a network which are mutually exclusive and under perfect competition i.e. if a user buys product 'A' (or gets infected with virus 'X'), she will never buy product 'B' (or virus 'Y'). This is not always true: for example, a user could install and use both Firefox and Google Chrome as browsers. Similarly, one type of flu may give partial immunity against some other similar disease.

In the case of full competition, we proved that 'winner takes all,' that is the weaker virus/product will become extinct. In the case of no competition, both viruses survive, ignoring each other. So a natural question is: what happens in-between these two extremes?

In this chapter, we show that there is a phase transition: if the competition is harsher than a critical level, then 'winner takes all;' otherwise, the weaker virus survives. Our contributions are: (a) the problem definition, which is novel even in epidemiology literature [AM91, Het00, Ste09] (b) the phase-transition result and (c) experiments on real data, illustrating the suitability of our results.

5.1 Introduction

Given two partially competing products (like Firefox and Google Chrome; or Android and iPhone), is it possible that they both survive?

The well-known Competitive Exclusion Principle in ecology states that when two species are in complete competition under constant conditions, the more fit one will eventually drive the less fit one into extinction. A more common but less well understood scenario is one where the competing species induce partial immunity against one another. There has been significant work trying to elucidate the conditions under which such partial immunity leads to coexistence [LCC⁺09, CCF⁺10, LSN96] but a complete theory has not yet emerged.

Here, we study the general case of two virus strains with partial (and symmetric) cross-immunity spreading over a fixed network topology. In addition to the implications

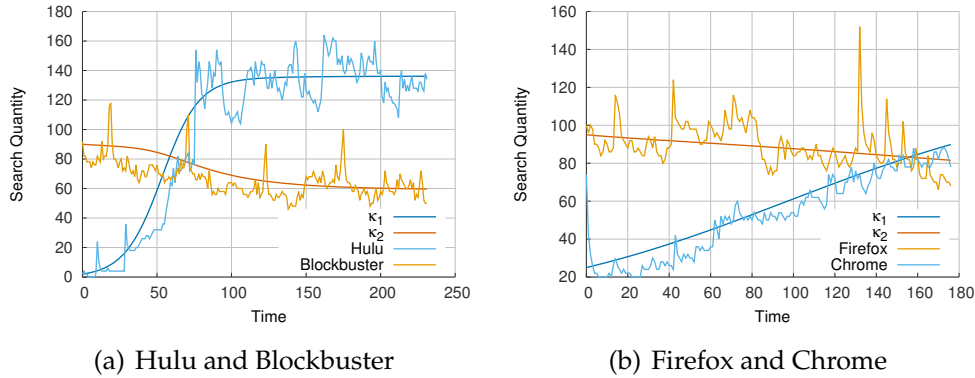


Figure 5.1: Plots of real web-search interest vs. time for pairs of competitors with our model fitted to the data.

for the evolutionary problem discussed above, our results have direct relevance to the spread of rumors and opinions in social networks and market penetration of products.

The contributions of this work are the following:

- the discovery that there is a phase transition, that is, the weaker virus/product may survive, if the cross-immunity satisfies a threshold condition. This seemed to be an open problem even within the epidemiology community [LCC⁺09]
- experiments on real data, showing that our model fits well

Figure 5.1 shows the time-plots for partially competing products Hulu vs. Blockbuster (a), and Google Chrome vs. Firefox (b). They plot (normalized) count of Google queries, versus time. We fit our model to the data¹ and plot it as well. Notice that it captures the trends well.

The rest of the chapter is organized in the usual way: we review related work in § 5.2 and formulate the problem giving details of our model in § 5.3. We give the analysis and proof of our phase-transition and coexistence result in § 5.4 and demonstrate the validity of the results using simulations and real-world case-studies in § 5.5. Finally, we discuss other subtle aspects of the model in § 5.6 and conclude in § 5.7.

5.2 Related Work

We give a very brief summary of the related work. For much of the prior work in this area please see the related work in the previous chapter. As mentioned before, most works deal with single viruses. Our previous chapter looked at a two-virus SIS model on arbitrary graphs, but focused on the case where there was full mutual immunity between viruses. The main result says that, in such a setting, the stronger virus will push the weaker one to extinction (‘winner takes all’), even if the weaker one would be able to survive on the network when left alone.

¹Fitted with www.alexbeutel.com/jsplot/kdd2012.html

Partial immunity models have received much attention in epidemiology. For example, [LCC⁺09] suggests a differential equation based model and analyzes it via simulation. However, for this and most other models of interest, a complete analytical solution has been beyond reach.

Distinguishing features of current work: In short, none of the previous work fulfills all the conditions of this current work: (a) analytical proof of $\epsilon_{\text{critical}}$, the critical value of the competition threshold (b) closed-form steady-state behavior (c) under an SIS (flu-like) model.

5.3 Problem Formulation

In this section, we formulate our problem, giving details about the model used and the assumptions. Table 5.1 explains the terminology we have used in the chapter. Bold letters typically denote matrices (**A**, **M** etc.).

Table 5.1: Symbols and Definitions

Symbol	Definition and Description
$SI_{1 2}S$	our competing viruses model
β_1 (or β_2)	attack rate of virus 1 (or virus 2)
δ_1 (or δ_2)	cure rate of virus 1 (or virus 2)
ϵ	Interaction factor between virus 1 and 2
A	adjacency matrix of the underlying graph
$\lambda_1(\mathbf{M})$	largest eigenvalue of matrix M
λ	$\lambda_1(\mathbf{A})$
σ_1	$\lambda\beta_1/\delta_1$ (strength of virus 1)
σ_2	$\lambda\beta_2/\delta_2$ (strength of virus 2)
I_1 (or I_2)	The number of nodes infected with only virus 1 (or only virus 2)
$I_{1,2}$	The number of nodes infected with both virus 1 and virus 2
κ_1	Fraction of nodes infected with virus 1 $((I_1 + I_{1,2})/N)$
κ_2	Fraction of nodes infected with virus 2 $((I_2 + I_{1,2})/N)$
i_{12}	Fraction of nodes infected with both viruses $(I_{1,2}/N)$
i_1 (or i_2)	Fraction of nodes infected with virus 1: I_1/N (or virus 2: I_2/N)
$\kappa_1^*, \kappa_2^*, i_{12}^*$	The solution at the coexistence equilibrium (if exists)

5.3.1 The propagation model

We assume that the competing viruses are spreading on the network according to a propagation model, which we describe next. We call our propagation model $SI_{1|2}S$, based

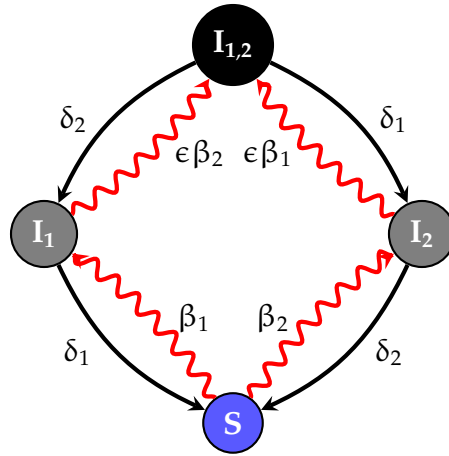


Figure 5.2: State Diagram for a node in the graph under our partial-competition model.

on the popular “flu-like” SIS (Susceptible-Infected-Susceptible) model [Het00]. $SI_{1|2}S$ denotes Susceptible - Infected_{1 or 2} - Susceptible. Each node in the graph can be in one of four states: Susceptible (healthy), I_1 (infected by virus 1), I_2 (infected by virus 2), or $I_{1,2}$ (infected by both virus 1 and virus 2). The state transition diagram as seen from a node in the network is shown in Figure 5.2. We could have extended other single virus models as well, but we believe that our model is a reasonable starting point, and we leave the analysis of other models as future work.

Healing (virus death) rate: δ . If a node is in an infected state (I_1 , I_2 , or $I_{1,2}$), it recovers on its own with some rate, δ_1 for virus 1 or δ_2 for virus 2. The healing rate is inversely related with the virus’s strength: a high δ means that nodes that are infected heal quickly. For example, a product that lasts a long time such that people using it rarely consider alternatives would be modeled with a low δ value.

Attack (virus transmission) rate: β . An infected node can spread the virus to its neighboring nodes, and the node susceptibility is captured by β_1 and β_2 . Specifically, an infected node transmits its infection to each of its healthy neighbors *independently* at rate β_1 (or β_2). The more often an idea or product is shared with friends, frequently referred to as being “viral,” the higher the value for β .

Virus interaction factor: ϵ . A node infected with one virus may be more or less susceptible to being infected by the other virus, as determined by the factor ϵ . The transmission rate for a virus becomes $\epsilon\beta_1$ (or $\epsilon\beta_2$) when a node is already infected with one virus. Specifically, if a node is infected with virus 1, each of its neighbors infected with virus 2 have a transmission rate to it of $\epsilon\beta_2$; a node infected with virus 2 can only be infected with virus 1 at a rate of $\epsilon\beta_1$.

This is a novel generalization of the single-virus SIS model to a multiple-virus scenario. The value of ϵ can describe many different virus interactions. If $\epsilon = 0$ then the viruses are fully mutually immune, and $0 < \epsilon \leq 1$ suggests an amount of competition between viruses.

Fair-play: We assume that the competitors are playing a ‘fair game’: All nodes in the network have the same model parameters (β 's, δ 's, ϵ) for each of the viruses and behave according to the state-diagram in Figure 5.2.

5.3.2 Problem Statement

We are now in a position to state the problem formally. We assume the underlying network is connected - otherwise we just have separate disconnected problems.

Interacting viruses problem

Given: An undirected connected graph G , and the propagation model ($SI_{1|2}S$) parameters (β_1, δ_1 for virus 1, β_2, δ_2 for virus 2, and ϵ)

Find: What are the possible fixed points for the system? In particular, for what values of ϵ is there a fixed point for which both virus 1 and virus 2 survive?

5.3.3 Model Formulation for a Clique

For a clique, the following differential equations fully describe the transitions of the system, seen in Figure 5.2. Here I_1 , I_2 , and $I_{1,2}$ are the number of nodes infected with only virus 1, only virus 2, and both virus 1 and 2 respectively. N is the total number of nodes, and S is the number of susceptible nodes ($S = N - I_1 - I_2 - I_{1,2}$).

$$\frac{dI_1}{dt} = \beta_1 S(I_1 + I_{1,2}) + \delta_2 I_{1,2} - \delta_1 I_1 - \epsilon \beta_2 I_1(I_2 + I_{1,2}) \quad (5.1)$$

$$\frac{dI_2}{dt} = \beta_2 S(I_2 + I_{1,2}) + \delta_1 I_{1,2} - \delta_2 I_2 - \epsilon \beta_1 I_2(I_1 + I_{1,2}) \quad (5.2)$$

$$\frac{dI_{1,2}}{dt} = \epsilon \beta_1 I_2(I_1 + I_{1,2}) + \epsilon \beta_2 I_1(I_2 + I_{1,2}) - (\delta_1 + \delta_2) I_{1,2} \quad (5.3)$$

5.4 Results and Proofs

The goal of our analysis is to find for what values of ϵ is there an equilibrium point for which both virus 1 and virus 2 survive. We find that there is an $\epsilon_{\text{critical}}$ such that if $\epsilon > \epsilon_{\text{critical}}$ then an equilibrium point for which the viruses coexist.

5.4.1 Formulating the problem

At an equilibrium point, all derivatives are zero. Thus, we can find a simple equation for $I_{1,2}$

$$\epsilon(\beta_1 + \beta_2)I_1I_2 = (\delta_1 + \delta_2 - \epsilon(\beta_1I_2 + \beta_2I_1))I_{1,2}$$

Lemma 5.1. *The number of people infected by both virus 1 and virus 2 will obey the following equation:*

$$I_{12} = I_1 I_2 \epsilon (\beta_1 + \beta_2) / (\delta_1 + \delta_2 - \epsilon (\beta_1 I_2 + \beta_2 I_1))$$

Proof. Trivial, given the above. □

Thus we have the expected three equilibrium points

- $I_1 = I_2 = I_{12} = 0$
- $I_1 = I_{12} = 0, I_2 = N - \frac{\delta_2}{\beta_2}$
- $I_2 = I_{12} = 0, I_1 = N - \frac{\delta_1}{\beta_1}$

and possibly one for which $I_1, I_2 > 0$ and obeys the differential equations outlined:

$$0 = \beta_1 S(I_1 + I_{12}) + \delta_2 I_{12} - \delta_1 I_1 - \epsilon \beta_2 I_1 (I_2 + I_{12}) \quad (5.4)$$

$$0 = \beta_2 S(I_2 + I_{12}) + \delta_1 I_{12} - \delta_2 I_2 - \epsilon \beta_1 I_2 (I_1 + I_{12}) \quad (5.5)$$

$$0 = \epsilon \beta_1 I_2 (I_1 + I_{12}) + \epsilon \beta_2 I_1 (I_2 + I_{12}) - (\delta_1 + \delta_2) I_{12} \quad (5.6)$$

We rework these equations to be primarily in terms of $\kappa_1, \kappa_2, i_{12}$, where $\kappa_1 = (I_1 + I_{12})/N$, $\kappa_2 = (I_2 + I_{12})/N$, $i_{12} = I_{12}/N$. As such, each of these terms represent a fraction of the population that is infected. We first convert the constraints to

$$N \kappa_1 \beta_1 [1 - \kappa_1 - (1 - \epsilon) i_{12}] = \delta_1 \kappa_1 \quad (5.7)$$

$$N \kappa_2 \beta_2 [1 - \kappa_2 - (1 - \epsilon) i_{12}] = \delta_2 \kappa_2 \quad (5.8)$$

$$\epsilon N (\beta_1 \kappa_1 i_{12} + \beta_2 \kappa_2 i_{12}) = (\delta_1 + \delta_2) i_{12} \quad (5.9)$$

where $i_1 = I_1/N$ and $i_2 = I_2/N$.

Manipulating (5.9) to remove i_1 and i_2 , we find

$$\epsilon \kappa_1 \kappa_2 [\sigma_1 \delta_1 + \sigma_2 \delta_2] = i_{12} [\delta_1 + \delta_2 + \epsilon \sigma_1 \delta_1 \kappa_1 + \epsilon \sigma_2 \delta_2 \kappa_2] \quad (5.10)$$

Remember, because we are working with a clique the virus strengths are $\sigma_1 = N \beta_1 / \delta_1$ and $\sigma_2 = N \beta_2 / \delta_2$.

5.4.2 Results

From these constraints, we look to find a lower bound on ϵ , such that for any less competition there can be coexistence.

Theorem 5.1 (Epsilon Threshold Theorem). *Given a fully connected graph with the $SI_{1|2}S$ model parameters $\sigma_1 \geq \sigma_2$, an equilibrium point for which $\kappa_1, \kappa_2 > 0$ exists if $\epsilon > \epsilon_{\text{critical}}$, where*

$$\epsilon_{\text{critical}} = \begin{cases} \frac{\sigma_1 - \sigma_2}{\sigma_2 (\sigma_1 - 1)} & \text{if } \sigma_1 + \sigma_2 \geq 2 \\ \frac{2(1 + \sqrt{1 - \sigma_1 \sigma_2})}{\sigma_1 \sigma_2} & \text{if } \sigma_1 + \sigma_2 < 2 \end{cases} \quad (5.11)$$

Proof. In Lemma 5.3 we give the possible fixed point for coexistence. In Lemma 5.4 we show the constraints for the fixed points to be real, which contribute to the bounds in (5.11). In Lemmas 5.5 through 5.9 we give the proofs for the constraints on the fixed points being positive, and in Lemma 5.10 we give the proof that the fixed points are less than one. \square

Next we describe all of the Lemmas, which contribute to the proof.

Lemma 5.2. *If a fourth equilibrium point exists, then it should satisfy the follow equation:*

$$\epsilon(\kappa_2 - \kappa_1) = 1/\sigma_1 - 1/\sigma_2 \quad (5.12)$$

Proof. Since we are only looking for non-zero solutions for κ_1 and κ_2 , we can eliminate them in (5.7) and (5.8).

$$1 - \kappa_1 - (1 - \epsilon)i_2 = 1/\sigma_1 \quad (5.13)$$

$$1 - \kappa_2 - (1 - \epsilon)i_1 = 1/\sigma_2 \quad (5.14)$$

Subtracting, we get the lemma. \square

Lemma 5.3 (Coexistence Lemma). *If an equilibrium point exists for which both viruses coexist in the network, $\kappa_1, \kappa_2 > 0$, it will be at:*

$$i_{12} = \epsilon\kappa_1\kappa_2 \left[\frac{\sigma_1\delta_1 + \sigma_2\delta_2}{\delta_1 + \delta_2 + \epsilon\sigma_1\delta_1\kappa_1 + \epsilon\sigma_2\delta_2\kappa_2} \right] \quad (5.15)$$

$$\kappa_1 = \kappa_2 + \frac{1}{\epsilon} \left(\frac{1}{\sigma_2} - \frac{1}{\sigma_1} \right) \quad (5.16)$$

$$\kappa_2 = \frac{-2\epsilon\sigma_1\sigma_2 + \epsilon^2\sigma_1\sigma_2^2 \pm \epsilon\sqrt{\sigma_1\sigma_2^3}\sqrt{4-4\epsilon+\epsilon^2\sigma_1\sigma_2}}{2\epsilon^2\sigma_1\sigma_2^2} \quad (5.17)$$

We will denote the solution to these three equations for fixed-points as i_{12}^* , κ_1^* , and κ_2^* respectively.

Proof. Equation (5.15) is a simple rearrangement of equation (5.10), and equation (5.16) is a rearrangement of equation (5.12). Plugging (5.15) and (5.16) into (5.13) allows us to solve for κ_2 resulting in (5.17).

For κ_2^* (and by extension κ_1^* and i_{12}^*) to be a valid fixed-point, κ_2^* must be: (a) real, (b) $\kappa_2^* \geq 0$, (c) $\kappa_2^* \leq 1$.

Lemma 5.4. *In order for fixed-point solution κ_2^* , and by extension κ_1^* and i_{12}^* , to be real valued, either $\sigma_1\sigma_2 > 1$ or*

$$\epsilon < \frac{2(1 - \sqrt{1 - \sigma_1\sigma_2})}{\sigma_1\sigma_2} \quad \text{or} \quad \epsilon > \frac{2(1 + \sqrt{1 - \sigma_1\sigma_2})}{\sigma_1\sigma_2}.$$

Proof. This constraint comes from the square root in equation (5.17) for κ_2^* . We analyze the quadratic equation $4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2$ (in terms of ϵ) from inside the square root. It is a simple, upward-facing parabola. Solving for the roots of the quadratic equation in terms of ϵ we find

$$\epsilon = \frac{2(1 \pm \sqrt{1 - \sigma_1\sigma_2})}{\sigma_1\sigma_2}.$$

For $\sigma_1\sigma_2 > 1$ there is no solution because the equation is positive for all values of ϵ . Thus, if $\sigma_1\sigma_2 > 1$ then κ_2^* must be real valued. For $\sigma_1\sigma_2 < 1$ a portion of the parabola is negative. Therefore, we require that ϵ be in the positive region of the quadratic equation, where ϵ is less than the lower root or greater than the upper root. \square

To find when $\kappa_2^* \geq 0$, we consider the cases above for which it is real. As we explained before, we will focus on the lower bound for ϵ .

Lemma 5.5. *For strengths $\sigma_1\sigma_2 > 1$, fixed-point κ_2^* is monotonically increasing as a function of ϵ .*

Proof. Taking the derivative of (5.17) we get

$$\frac{\pm(-2 + \epsilon)\sqrt{\sigma_2} + \sqrt{\sigma_1}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2}}{\epsilon^2\sqrt{\sigma_1\sigma_2}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2}}.$$

Because $\sigma_1\sigma_2 > 1$, all of the square roots are real valued. The denominator is clearly positive, so to prove that κ_2^* is monotonically increasing, we must show that the numerator is positive. To show that the numerator is always positive we would like to show that

$$\pm(-2 + \epsilon)\sqrt{\sigma_2} < \sqrt{\sigma_1}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2}$$

or alternatively

$$1 < \frac{\sigma_1}{\sigma_2} \frac{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2}{4 - 4\epsilon + \epsilon^2}.$$

Because $\sigma_1 \geq \sigma_2$ the first term is clearly > 1 . For $\sigma_1\sigma_2 > 1$ (and of course $\epsilon > 0$) this is trivially true. \square

Lemma 5.6. *Fixed-point solution κ_2^- , defined by*

$$\kappa_2^- = \frac{-2\epsilon\sigma_1\sigma_2 + \epsilon^2\sigma_1\sigma_2^2 - \epsilon\sqrt{\sigma_1}\sigma_2^{3/2}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2}}{2\epsilon^2\sigma_1\sigma_2^2}, \quad (5.18)$$

can only be positive when κ_2^+ , defined by

$$\kappa_2^+ = \frac{-2\epsilon\sigma_1\sigma_2 + \epsilon^2\sigma_1\sigma_2^2 + \epsilon\sqrt{\sigma_1}\sigma_2^{3/2}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2}}{2\epsilon^2\sigma_1\sigma_2^2}, \quad (5.19)$$

is positive.

Proof. As a simple case, for $\sigma_1\sigma_2 > 1$, $\kappa_2^- < 0$ and thus invalid for all $\epsilon > 0$. As ϵ approaches 0, it is clear that $\kappa_2^- \rightarrow -\infty$, and as $\epsilon \rightarrow \infty$, we see that κ_2^- approaches 0. Since from the previous lemma we know that it is monotonically increasing, $\kappa_2^- < 0$ for $\sigma_1\sigma_2 > 1$.

If we do not restrict σ_1 and σ_2 , it is still clear that $\kappa_2^- < \kappa_2^+$ for all $\epsilon \geq 0$, since the last term is always positive. We will see later that $\kappa_2^+ < 1$ for all $\epsilon > 0$. Therefore, the range for which κ_2^- is valid is a strict subset of that for which κ_2^+ is valid. \square

Because κ_2^- is only valid when κ_2^+ is valid, it has no impact on the phase transition claimed in Theorem 5.1. As a result, we will focus on κ_2^+ for the remainder of the proof and, with a slight abuse of notation, use κ_2^* to denote κ_2^+ .

Lemma 5.7. *When strengths $\sigma_1\sigma_2 \geq 1$, the fixed-point for the population infected by virus 2 is positive, $\kappa_2^* > 0$, if and only if*

$$\epsilon > \frac{\sigma_1 - \sigma_2}{\sigma_2(\sigma_1 - 1)}.$$

Proof. Solving equation (5.19) for $\kappa_2^* = 0$ produces $\epsilon = \frac{\sigma_1 - \sigma_2}{\sigma_2(\sigma_1 - 1)}$. Because κ_2^* is monotonically increasing in this region ($\sigma_1\sigma_2 > 1$), for all ϵ greater than this solution, $\kappa_2^* > 0$, and for all ϵ less than this solution $\kappa_2^* \leq 0$. \square

Lemma 5.8. *If virus strengths $\sigma_1 + \sigma_2 < 2$, then the fixed-point for the population infected by virus 2 is positive, $\kappa_2^* > 0$, for*

$$\epsilon > \frac{2(1 + \sqrt{1 - \sigma_1\sigma_2})}{\sigma_1\sigma_2}.$$

Proof. For κ_2^* to be positive, the numerator of (5.19) must be positive. We can reduce this as follows:

$$\begin{aligned} & -2\epsilon\sigma_1\sigma_2 + \epsilon\sigma_1\sigma_2^2 + \epsilon\sqrt{\sigma_1}\sigma_2^{3/2}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2} \\ & = \epsilon\sigma_2(\sqrt{\sigma_1\sigma_2}\sqrt{4 - 4\epsilon + \epsilon^2\sigma_1\sigma_2} - 2\sigma_1 + \epsilon\sigma_1\sigma_2) \\ & \geq \epsilon\sigma_2(0 - 2\sigma_1 + 2(1 + \sqrt{1 - \sigma_1\sigma_2})) \\ & = 2\epsilon\sigma_2(-\sigma_1 + 1 + \sqrt{1 - \sigma_1\sigma_2}) \end{aligned}$$

For this to be positive we must have $\sqrt{1 - \sigma_1\sigma_2} > \sigma_1 - 1$, which is true for $\sigma_1 + \sigma_2 < 2$. \square

Lemma 5.9. *If virus strengths $\sigma_1 + \sigma_2 \geq 2$ then the fixed-point for the population infected by virus 2 is positive, $\kappa_2^* > 0$, for*

$$\epsilon > \frac{\sigma_1 - \sigma_2}{\sigma_2(\sigma_1 - 1)}.$$

Proof. Again, for κ_2^* to be positive, the numerator of (5.19) must be positive. We can reduce this as follows:

$$\begin{aligned}
& -2\epsilon\sigma_1\sigma_2 + \epsilon\sigma_1\sigma_2^2 + \epsilon\sqrt{\sigma_1}\sigma_2^{3/2}\sqrt{4-4\epsilon+\epsilon^2\sigma_1\sigma_2} \\
& = \epsilon\sigma_2(\sqrt{\sigma_1\sigma_2}\sqrt{4-4\epsilon+\epsilon^2\sigma_1\sigma_2} - 2\sigma_1 + \epsilon\sigma_1\sigma_2) \\
& \geq \epsilon\sigma_2\left(\sqrt{\sigma_1\sigma_2}\sqrt{\frac{\sigma_1(-2+\sigma_1+\sigma_2)^2}{\sigma_2(-1+\sigma_1)^2}} - 2\sigma_1 + \sigma_1\sigma_2\left(\frac{\sigma_1-\sigma_2}{\sigma_2(1-\sigma_1)}\right)\right) \\
& = \epsilon\sigma_1\sigma_2\left(\frac{2-\sigma_1-\sigma_2}{1-\sigma_1} - 2 + \frac{\sigma_1-\sigma_2}{1-\sigma_1}\right) = 0
\end{aligned}$$

□

Lemma 5.10. *The fixed-point for the population infected by virus 2 is valid, $\kappa_2^* \leq 1$, for $\sigma_1 \geq \sigma_2$ and $\epsilon \geq 0$.*

Proof. The constraint $\kappa_2^* \leq 1$ is equivalent to

$$-2\epsilon\sigma_1\sigma_2 - \epsilon^2\sigma_1\sigma_2^2 + \epsilon\sqrt{\sigma_1}\sigma_2^{3/2}\sqrt{4-4\epsilon+\epsilon^2\sigma_1\sigma_2} < 0.$$

This can be simplified as follows:

$$\sqrt{\sigma_1\sigma_2}\sqrt{4-4\epsilon+\epsilon^2\sigma_1\sigma_2} < 2\sigma_1 + \epsilon\sigma_1\sigma_2 \quad (5.20)$$

$$\sigma_1\sigma_2(4-4\epsilon+\epsilon^2\sigma_1\sigma_2) < 4\sigma_1^2 + \epsilon^2\sigma_1^2\sigma_2^2 + 4\epsilon\sigma_1^2\sigma_2 \quad (5.21)$$

$$\sigma_1\sigma_2 - \epsilon\sigma_1\sigma_2 < \sigma_1^2 + \sigma_1^2\sigma_2 \quad (5.22)$$

$$\frac{\sigma_2}{\sigma_1} \frac{1-\epsilon}{1+\epsilon\sigma_2} < 1 \quad (5.23)$$

The simplification to (5.23) makes it clear that the lemma is true for $\sigma_1 \geq \sigma_2 > 0$. □

As such, for any interaction factor $\epsilon > \epsilon_{\text{critical}}$, we have proved that κ_1^* and κ_2^* are valid equilibrium points for which the population infected by each virus $\kappa_1, \kappa_2 > 0$. □

□

5.5 Experiments

We demonstrate our result using (a) simulation experiments and (b) case studies using real data in this section.

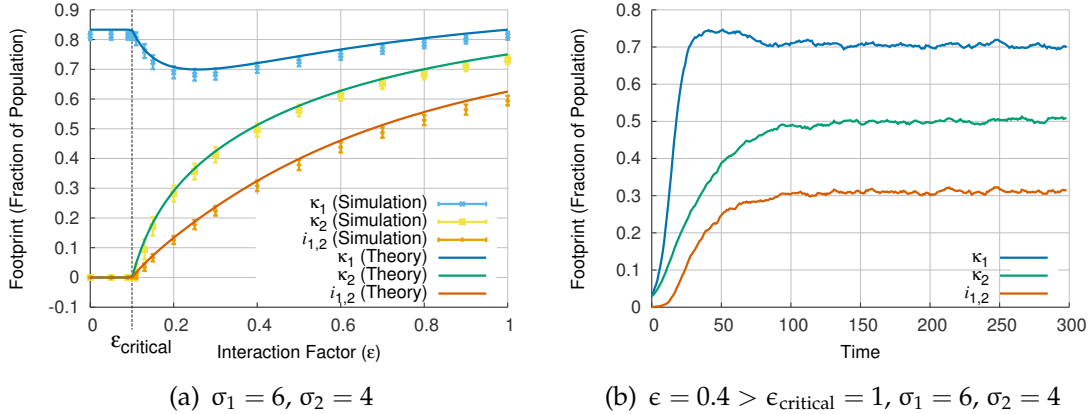


Figure 5.3: Coexistence is possible: Results from simulations on clique of size 1000 with theoretical fixed points overlaid. (a) shows the steady-state population values, κ_1 , κ_2 , and $i_{1,2}$, for each value of ϵ , with the theoretical $\epsilon_{\text{critical}}$ marked. (b) shows the development of the two viruses over time for $\epsilon > \epsilon_{\text{critical}}$. Notice that both viruses survive as expected.

5.5.1 Setup

Without loss of generality, in our experiments we assumed that the first virus is the stronger virus. We primarily focus on the case where $\sigma_1 > \sigma_2 > 1$. For our simulations we use $\sigma_1 = 6$ and $\sigma_2 = 4$.

We run a simulation on a fully-connected clique of 1000 nodes. We vary ϵ around our expected threshold and for each value of ϵ perform 10 runs over 4000 time steps. On each run we begin by infecting 30 nodes at random with each virus.

We analyze the results in two ways. First, we create a *steady-state plot* of mean values and standard deviations for κ_1 , κ_2 , and $i_{1,2}$ at steady-state over a range of values for ϵ . Over the results of the simulation we draw the behavior predicted by our results. Second, for one $\epsilon > \epsilon_{\text{critical}}$ we track each virus's development over time with a *time-plot*. The time-plot takes the average number of nodes infected (κ_1 , κ_2 , and $i_{1,2}$) at each time step and plots this against time. Although the simulations were run for 4000 time steps, the plots are truncated to give more detail to the initial fluctuations of the virus counts.

5.5.2 Simulation Results

Figure 5.3 displays our results. In short, the plots agree exactly with our result, as expected. Figure 5.3(a) shows the steady-state plot for the two viruses, and the theoretical predictions closely match the simulation results. Similarly, the viruses' growth as shown in the time-plot in Figure 5.3(b) matches what is expected.

For the steady-state plot, we expect the steady-state value to be one of the other fixed points where at least one virus dies out for $\epsilon \leq \epsilon_{\text{critical}}$ and then a co-existence for $\epsilon > \epsilon_{\text{critical}}$. In Figure 5.3(a) we see for $\sigma_1 > \sigma_2 > 1$ and $\epsilon = 0$, winner takes all, as was

proven in [PBRF12]. However, for $\epsilon > \epsilon_{\text{critical}}$ we see a coexistence between the viruses as expected; this is true even when the viruses are competing ($\epsilon < 1$). For $\epsilon = 0.4 > \epsilon_{\text{critical}}$ we have the time-plot, Figure 5.3(b), showing the growth of both viruses to steady-state from a small infection in the system.

5.5.3 Case-Studies using Real Data

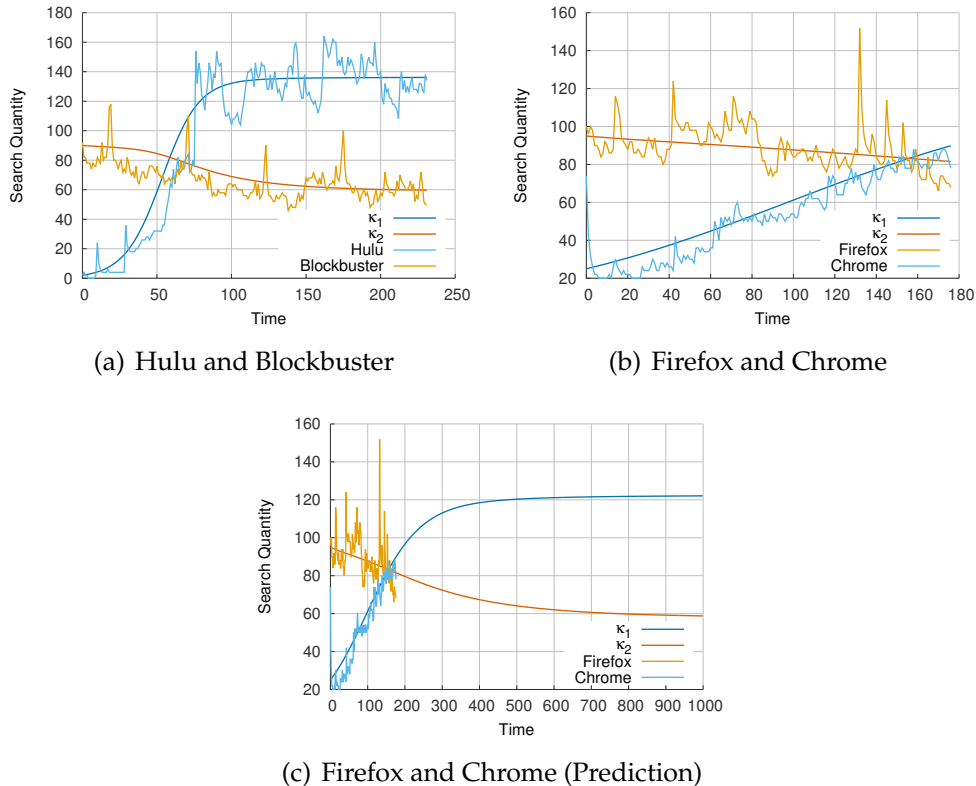


Figure 5.4: Real web-search interest vs. time plots for pairs of competitors with our model fitted to it. (c) Predicts steady state values based on our model. Data acquired from Google Trends.

We collected historical data for ‘web-search interest’ for various competing products from the Google-Insights website², which aims to “provide insights into broad search patterns.” This allows us to use the data as a proxy for sales/interest for each product. We used the following pairs of rival products:

1. Hulu³ and Blockbuster⁴: Although not direct competitors, both offer video entertainment services, though under very different models.

²www.google.com/insights/search/

³www.hulu.com

⁴www.blockbuster.com

2. Firefox⁵ and Google Chrome⁶: Two rival web browsers.

We consider both pairs of products to be examples of cases where there is partial mutual immunity; people can use both products, but the use of one we expect would detract from the use of the other. While our model does not describe the situations perfectly, we believe it is a good approximation.

In Figure 5.4 we show plots of the web-interest vs. time for both pairs of products, along with our model fitted to the data⁷. In Figure 5.4(a), we used a virus interaction factor of $\epsilon = 0.7$ (along with virus parameters $\delta_{\text{Hulu}} = 0.04$, $\beta_{\text{Hulu}} = 0.0007$, $\delta_{\text{Blockbuster}} = 0.05$, $\beta_{\text{Blockbuster}} = 0.00045$). In Figure 5.4(b), we used a virus interaction factor of $\epsilon = 0.6$ (along with virus parameters $\delta_{\text{Firefox}} = 0.01$, $\beta_{\text{Firefox}} = 0.000095$, $\delta_{\text{Chrome}} = 0.01$, $\beta_{\text{Chrome}} = 0.00015$). In Figure 5.4(c) we use the same model as (b) but let the model continue to see the projected steady state behavior. We note that the plots begin when Hulu and Chrome are first introduced and with Blockbuster and Firefox at a previous steady-state behavior. In each of these fittings we see that our model fits the data well. The fact that the model fits the data well demonstrates the suitability of our $SI_{1|2}S$ model.

5.6 Discussion

5.6.1 A general upper bound

Conjecture 5.1 (Epsilon Threshold Upper Bound). *Given an arbitrary graph with the $SI_{1|2}S$ model parameters $\sigma_1 \geq \sigma_2 \geq 1$, an equilibrium point for which both virus 1 and virus 2 survive exists if $\epsilon > \epsilon_{\text{critical}}$, where*

$$\epsilon_{\text{critical}} \leq \frac{1}{\sigma_2} \quad (5.24)$$

Justification: Since $\sigma_1 \geq \sigma_2 \geq 1$ and $0 < \epsilon < 1$, we know that both virus 1 and virus 2 would be strong enough to survive independently but there is some competition between them. Because of the competition, as virus 1 spreads to more nodes, virus 2's attack rate on average decreases and thus its strength decreases. Therefore, if we overestimate the strength or number of people with virus 1, this only makes it more difficult for virus 2 survive and thus decreases the maximum amount of competition virus 2 can handle, increasing $\epsilon_{\text{critical}}$. To simplify the problem, we assume that *every person* is infected with virus 1 (as if virus 1 was infinitely strong). In this case, a node can only be in state I_1 or $I_{1,2}$, and the probability of a node with virus 2 infecting a neighbor is always $\epsilon\beta_2$. This is now equivalent to a one virus model where the strength of virus 2 is $\sigma'_2 = \epsilon\sigma_2$. Therefore, as shown in previous research [CWW⁺08, GMT05, PCF⁺11], if $\sigma'_2 > 1$ then virus 2 will

⁵www.mozilla.org/en-US/firefox/new/

⁶www.google.com/chrome

⁷Fitted with www.alexbeutel.com/jsplot/kdd2012.html

survive. In this case, $\epsilon_{\text{critical}} = \frac{1}{\sigma_2}$. However, because this is a relaxation of the original problem, we know that this is an upper bound and in fact $\epsilon_{\text{critical}} \leq \frac{1}{\sigma_2}$.

5.6.2 Case-Study: Qualitative Analysis

We also consider the example of educational ideas, and specifically sex education, as a virus. Sociology literature analyzing the success of sex education programs notes the impact of network effects and social structure on sex education success [BB05]. We match education policy to our $SI_{1|2}S$ model and analyze the implications.

Policy 1: Abstinence-Only Education Abstinence-only education teaches abstinence until marriage as the only way to live a healthy life, with students often taking an abstinence pledge. Under our model, virus 1 is believing in abstinence (through education or pledge) and virus 2 is sexual activity. Therefore, those who are in I_1 have taken a pledge of abstinence, those who are in I_2 are sexually active, and those people who are in S do not believe in abstinence but are not sexually active either. It is obviously impossible to both be following an abstinence pledge and to be sexually active so nobody can be in state $I_{1,2}$. Equivalently in this case there is full mutual immunity or $\epsilon = 0$.

Model 1 Predictions and Results Based on this fit, because $\epsilon = 0$, our model predicts ‘winner takes all:’ the weaker virus dies out and the stronger virus survives. Sociology research [BB05], studying over 11,000 people over 7 years, notes that of the 2399 people claiming to have taken an abstinence pledge, 1622 (67%) over time forgot. This suggests that $\sigma_{\text{Abstinence}} < \sigma_{\text{Sexual Activity}}$, and as a result, in the long run sexual activity will win over abstinence.

Policy 2: Comprehensive Sex Education Comprehensive sex education teaches numerous methods to have a safe, healthy sex life, discussing both contraception and abstinence. Matching this to our model, virus 1 is being educated in safe-sex practices and valuing their importance and virus 2 is sexual activity. Therefore, those who are in I_1 have been educated about safe-sex practices and believe they are important but are not sexually active, those in I_2 are sexually active but do not practice safe sex, those in $I_{1,2}$ practice safe-sex, and those in S are neither educated on safe-sex practices nor sexually active. Here we expect little to no competition between the two viruses and thus have an ϵ value close to, if not equal to, 1.

Model 2 Predictions and Results Because ϵ is close to 1, we expect that $\epsilon > \epsilon_{\text{critical}}$. As a result, it is possible for there to be coexistence of the two viruses, such that there can be a steady-state in which people are sexually active and practice safe-sex. This appears to match sociology literature claiming that those who initially use condoms will keep using condoms [BB05].

In summary, our model qualitatively agrees with sociology research and offers a plausible explanation for the results of the study. Additionally, these two cases demonstrate the value of a phase transition. In the first case, the model suggests winner takes all and the ineffectiveness of abstinence-only education. On the contrary, for policy 2, the model

predicts coexistence, which agrees with the findings, and is better for society.

5.6.3 Subtle Points

There are several subtle points, that we deferred until now, for clarity of exposition. Specifically, here we discuss the following issues:

What does it mean for $\epsilon > 1$?

As before, the virus 2 transmission rate for a virus infected with virus 1 becomes $\epsilon\beta_2$ and the virus 1 transmission rate for a virus infected with virus 2 becomes $\epsilon\beta_1$. However, because $\epsilon > 1$ the transmission rate for each virus increases for neighbors that are already infected. We consider this to be a form of cooperation between the viruses (products, ideas, etc.).

This pattern of cooperation between products is common in product ecosystems. An example of this is that people who have an iPod are more likely to buy music and videos through Apple's iTunes. Making use of such cooperation can be seen in 'freebie marketing' or the 'razor and blades business model,' in which the company producing razor blades sells the razors at an artificially low price creating a market for the blades. This method of tightly integrating products is common in a variety of industries.

What happens if σ_2 or $\sigma_1 \leq 1$?

Because $\sigma_1 \geq \sigma_2$ there are two cases we can analyze. The first is when $\sigma_1 \geq 1 > \sigma_2$, in which the second virus is too weak to survive on its own. We will refer to this as the "piggyback setting," because virus 2 can only survive with the help of the first. The second condition is when $1 > \sigma_1 \geq \sigma_2$, where both viruses are independently too weak to survive. We will refer to this as the "teamwork setting," because only through cooperation can both viruses survive. In each of these cases, Theorem 5.1 still holds - plugging in σ_1 and σ_2 we find an $\epsilon_{\text{critical}}$ for which a fixed point in which $\kappa_1, \kappa_2 > 0$ exists. This suggests that even if both viruses are independently too weak to survive on their own, with enough cooperation they can.

To test our theorem, we ran similar simulations where either one or both viruses were too weak to independently survive. In Figure 5.5(a) we show the steady-state plot for the piggyback case of $\sigma_1 > 1 > \sigma_2$. As expected, for $\epsilon = 1$ when the viruses are independent, virus 1 survives but virus 2 is not strong enough and dies out. For a sufficient amount of cooperation, $\epsilon > \epsilon_{\text{critical}}$, we find that virus 2 can survive as well. We see in Figure 5.5(c), the corresponding time-plot where $\epsilon = 3.5 > \epsilon_{\text{critical}}$, that once virus 1 grows, virus 2 is able to survive as well.

We also simulated the teamwork case, where neither virus is independently strong enough to survive, $1 > \sigma_1 \geq \sigma_2$. In Figure 5.5(b) we show the steady-state plot for this case. Again, the theoretical result and predicted phase-transition match the simulation

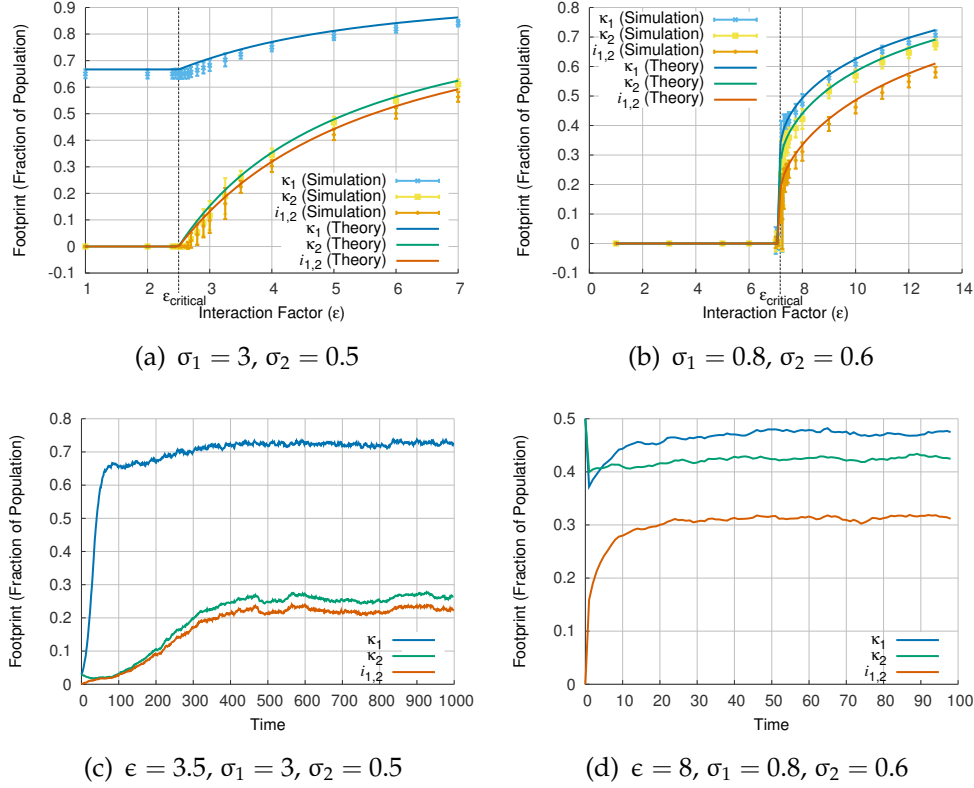


Figure 5.5: With enough collaboration, even weak viruses can survive. Results from simulations on population of $N = 1000$ with theoretical values overlayed, and $1 > \sigma_2$. The first column, (a) and (c), is the “piggyback” case where just virus 2 is too weak to survive. The second column, (b) and (d), is the “teamwork” case where neither virus is on its own strong enough to survive. The top row gives the steady-state plots, showing the steady-state footprint vs. the interaction factor ϵ . The second row gives the time-plots, showing the infection footprint developing over time.

results. For $\epsilon = 1$, the two viruses are independent and, since they cannot survive on their own, die out. In this case, the phase transition is based on the second part of Theorem 5.1 where $\sigma_1 + \sigma_2 < 2$, and as such the bound is a result on the restriction of κ_2 being real. As such, at $\epsilon_{\text{critical}}$ both $\kappa_1, \kappa_2 > 0$ rather than equal to 0. We see here at the threshold a large amount of uncertainty in the simulation but as we move away from the threshold the simulation follows this new fixed point. Interestingly, we must initially infect a large portion of the graph for the system to go to this fixed point, and not die out. For $\epsilon = 7.75 > \epsilon_{\text{critical}}$, Figure 5.5(d) shows the time-plot, demonstrating that both viruses quickly reach steady-state with a high amount of overlap.

5.7 Conclusions

We defined and studied the problem of partial competition, where two viruses/products provide partial immunity against each other.

The main contributions of our work are as follows:

1. *Problem Definition*: The problem is novel, in the data mining and web mining communities, and even in the epidemiology literature [AM91, Het00, Ste09].
2. *Threshold Result and Proof*: We showed that there is a phase transition: 'winner takes all,' until the competition level drops below a critical value. Above this critical value we find a closed-form steady-state solution with coexistence.
3. *Experiments and Case-studies*: We showed results from real settings (like browsers - Firefox vs Google Chrome), which agree with our model.

Part II
Algorithms

Overview

We turn our attention to *managing* cascade-like processes on networks. As we will see, these problems are closely related to the epidemic threshold problems we addressed in the previous part of this thesis. We solve two major problems here:

- **Effective and Efficient Immunization:** Given a large graph, like a computer communication network, which k nodes should we immunize (or monitor, or remove), to make it as robust as possible against a computer virus attack? Which nodes should we immunize to prevent an epidemic, if the virus is spreading over an underlying network changing over time (say day vs night connectivity)? How to distribute a fixed amount of infection-control resource (say antibiotics) amongst hospitals? We answer these and several other variants of the immunization problem. Our algorithms such as NETSHIELD and SMART-ALLOC achieve orders of magnitude improvement over current practice and other heuristics for complete immunization (aka node-removal) and fractional immunization over real hospital patient-transfer graphs, respectively. Finally, we also present effective algorithms when the underlying network changes with time.
- **Finding Culprits:** Next, we tackle a different and difficult question: Given a single snapshot of a partly infected network, how we can reliably identify those nodes from which the epidemic started; whether for inoculation to prevent future epidemics, or for exploitation for viral marketing. We propose to employ the Minimum Description Length principle for identifying that set of seed nodes from which the given snapshot can be described most succinctly. We introduce NETSLEUTH (based on a novel ‘submatrix-laplacian’ method), a highly efficient algorithm for both identifying the set of seed nodes that best describes the given situation, and *automatically* selecting the best number of seed nodes—in contrast to the state of the art.

These are very important tasks with numerous applications in a variety of areas, like public health (vaccination campaigns), cyber security (patching workstations), and online social-media analysis (detecting sources of rumors).

Chapter 6

Complete Node-Removal

Given a large graph, like a computer communication network, which k nodes should we immunize (or monitor, or remove), to make it as robust as possible against a computer virus attack? Which nodes should we immunize to prevent an epidemic, if the virus is spreading over an underlying network changing over time (say day vs night connectivity)? We need (a) a measure of the ‘*Vulnerability*’ of a given network (or the time-varying network), (b) a measure of the ‘*Shield-value*’ of a specific set of k nodes and (c) a fast algorithm to choose the best such k nodes.

It turns out that this problem is closely related to the epidemic threshold problem we studied in the preceding part of this thesis. Specifically, we answer all the above three questions: we give the justification behind our choices, we show that they agree with intuition as well as our previous results on the epidemic threshold (see Chapters 2 and 3). Moreover, we propose NETSHIELD, a fast and scalable algorithm which is provably *near-optimal* (within constant factor from optimal), when the network is static. We also give experiments on large real graphs, where NETSHIELD achieves tremendous speed savings exceeding *7 orders of magnitude*, against straightforward competitors. Finally, for the dynamic case, we give efficient heuristics and evaluate the effectiveness of our methods on synthetic and real data like the MIT reality mining graphs.

6.1 Introduction

Given a graph, we want to quickly find the k best nodes to immunize (or, equivalently, remove), to make the remaining nodes to be most robust to the virus attack. This is the core problem for many applications: In a computer network intrusion setting, we want the k best nodes to defend (e.g., through expensive and extensive vigilance), to minimize the spread of malware. Similarly, in a law-enforcement setting, given a network of criminals, we want to neutralize those nodes that will maximally scatter the graph. Similarly, which nodes should we immunize if the underlying network varies with time (motivated by the day-night pattern of human behavior)?

The problem boils down to three questions, which we address in this chapter. For the static case (similar questions can also be asked in the dynamic case of time-varying graphs.):

- Q1. *Vulnerability measure*: How to capture the ‘*Vulnerability*’ of the graph, in a single number? That is, how likely/easily that a graph will be infected by a virus. Similarly, how to capture the ‘*Vulnerability*’ of a series of graphs?
- Q2. ‘*Shield-value*’: How to quantify the ‘*Shield-value*’ of a given *set* of nodes in the graph, i.e., how important are they in terms of maintaining the ‘*Vulnerability*’ of the graph? Alternatively, how much less vulnerable will be the graph to the virus attack, if those nodes are immunized/removed?
- Q3. *Algorithm*: How to quickly determine the k nodes that *collectively* exhibit the highest ‘*Shield-value*’ score on large, disk-resident graphs?

When the graph does not change, we start by adopting the first eigenvalue λ of the graph as the ‘*Vulnerability*’ measurement (for Q1), which is motivated from immunology (based on our results in the previous chapters) and graph loop/path capacity. Based on that, we propose a novel definition of the ‘*Shield-value*’ score $S_v(\mathcal{S})$ for a specific set of nodes (for Q2). By carefully using the results from the theory of matrix perturbation, we show that the proposed ‘*Shield-value*’ gives a good approximation of the corresponding eigen-drop (i.e., the decrease of the ‘*Vulnerability*’ measurement if we remove/ immunize the set of nodes \mathcal{S} from the graph). Furthermore, we show that the proposed ‘*Shield-value*’ score is *sub-modular*, which enables us to develop a *near-optimal* and *scalable* algorithm (NETSHIELD) to find a set of nodes with highest ‘*Shield-value*’ score (for Q3). We conduct extensive experiments on several real data sets, illustrating the effectiveness and efficiency of the proposed methods. Specifically, our experiments show that the proposed method NETSHIELD (a) leads an effective immunization strategy; (b) scales *linearly* with the size of the graph; and (c) is dramatically faster than competitors (over 7 orders of magnitude).

Similarly, we choose the first eigenvalue of the system matrix λ_s (see Chapter 3) as the quality metric in the dynamic case. We further give efficient and effective heuristics to optimize it and demonstrate their efficacy through several experiments.

The rest of the chapter is organized as follows: Section 6.2 gives the related work. Sections 6.3-Section 6.7 deal with the problem in the static case (when the network does not change with time). Section 6.3 gives the problem definitions, while we present the ‘*Vulnerability*’ measurement in Section 6.4. The proposed ‘*Shield-value*’ score is presented in Section 6.5. We address the computational issues in Section 6.6. We evaluate the proposed methods in Section 6.7. We next tackle the immunization problem under the time-varying network case in Section 6.8. Finally, Section 6.9 gives the conclusions.

6.2 Related Work

In this section, we review the related work, which can be categorized into three parts: measuring the importance of nodes on graphs, immunization, and spectral graph analysis. Related work on general graph mining can be found in earlier chapters.

Measuring Importance of Nodes on Graphs In the literature, there are a lot of node importance measurements, including betweenness centrality, both the one based on the shortest path [Fre77] and the one based on random walks [New05b], PageRank [PBMW98], HITS [Kle98], and coreness score (defined by k-core decomposition) [MW03]. Other remotely related works include the abnormality score of a given node [SQCF05], articulation points [HT08], and k-vertex cut [HT08]. Our ‘*Shield-value*’ score is *fundamentally* different from these node importance scores, in the sense that they *all* aim to measure the importance of an individual node; whereas our ‘*Shield-value*’ tries to *collectively* measure the importance of a set of k nodes. Despite the fact that all these existing measures are successful for the goal they were originally designed for, they are not designed for the purpose of immunization. Therefore, it is not surprising that they lead to sub-optimal immunization results (See figure 6.4). Moreover, several of these importance measurements do not scale up well for large graphs, being cubic or quadratic wrt the number of nodes n , even if we use approximations (e.g., [MW08]). In contrast, the proposed NETSHIELD is linear wrt the number of edges and the number of nodes ($O(nk^2 + m)$). Another remotely related work is outbreak detection [LKG⁺07] in the sense that both works aim to select a subset of “*important*” nodes on graphs. However, the motivating applications (e.g., immunization) of this work is *different* from detecting outbreak [LKG⁺07] (e.g., contaminants in water distribution network). Consequently we solve *different* optimization problems (e.g., maximize the ‘*Shield-value*’ in eq. (6.2)) in this chapter.

Immunization Briesemeister et al. [BLP03] focus on immunization of power law graphs. They focus on the random-wiring version (that is, standard preferential attachment), versus the “highly clustered” power law graphs. Their simulation experiments on such synthetic graphs show that such graphs can be more easily defended against viruses, while random-wiring ones are typically overwhelmed, despite identical immunization policies.

Cohen et al. [CHbA03] studied the *acquaintance* immunization policy (see Section 6.8 for a description of this policy), and showed that it is much better than random, for both the SIS as well as the SIR model. For power law graphs (with no rewiring), they also derived formulae for the critical immunization fraction, above which the epidemic is arrested. Madar et al. [MKC⁺04] continued along these lines, mainly focusing on the SIR model for scale-free graphs. They linked the problem to bond percolation, and derived formulae for the effect of several immunization policies, showing that the “acquaintance” immunization policy is the best. Both works were analytical, without studying any real graphs.

Hayashi et al. [HMM03] study the case of a growing network, and they derive analytical formulas for such power law networks (no rewiring). They introduce the SHIR model (Susceptible, Hidden, Infectious, Recovered), to model computers under e-mail virus attack and derive the conditions for extinction under random and under targeted immunization, always for power law graphs with no rewiring.

In short, none of the above papers solves the problem of optimal immunization for an arbitrary, given graph or a set of arbitrary time-varying graphs.

Spectral Graph Analysis Pioneering works in this aspect can be traced back to Fiedler’s seminal work [Fie73]. Representative follow-up works include [SM97, NJW01, ZHD⁺01, DLJ08], etc. All of these works use the eigen-vectors of the graph (or the graph Laplacian) to find communities in the graph.

6.3 Problem Definitions (Static Graphs)

Table 6.1 lists the main symbols we use throughout the chapter. In this work, we focus on un-directed un-weighted graphs. We represent the graph by its adjacency matrix. Following standard notations, we use capital bold letters for matrices (e.g., \mathbf{A}), lower-case bold letters for vectors (e.g., \mathbf{a}), and calligraphic fonts for sets (e.g., \mathcal{S}). We denote the transpose with a prime (i.e., \mathbf{A}' is the transpose of \mathbf{A}), and we use parenthesized superscripts to denote the corresponding variable after deleting the nodes indexed by the superscripts. For example, λ is the first eigen-value of \mathbf{A} , then λ^i is the first eigen-value of \mathbf{A} after deleting its i^{th} row/column. We use $(\lambda_i, \mathbf{u}_i)$ to denote the i^{th} eigen-pair (sorted by the magnitude of the eigenvalue) of \mathbf{A} . When the subscript is omitted, we refer to them as the first eigenvalue and eigenvector respectively (i.e., $\lambda \triangleq \lambda_1$ and $\mathbf{u} \triangleq \mathbf{u}_1$).

With the above notations, our problems can be formally defined as follows:

Problem 6.1. Measuring ‘Vulnerability’

Given: A large un-directed un-weighted connected graph G with adjacency matrix \mathbf{A} ;

Find: A single number $V(\mathbf{G})$, reflecting the ‘Vulnerability’ of the whole graph.

Problem 6.2. Measuring ‘Shield-value’

Given: A subset \mathcal{S} with k nodes in a large un-directed un-weighted connected graph \mathbf{A} ;

Find: A single number $Sv(\mathcal{S})$, reflecting the ‘Shield-value’ of these k nodes (that is, the benefit of their removal/immunization to the vulnerability of the graph).

Problem 6.3. Finding k Nodes of Best ‘Shield-value’

Given: A large un-directed un-weighted connected graph \mathbf{A} with n nodes and an integer k ;

Find: A subset \mathcal{S} of k nodes with the highest ‘Shield-value’ score among all $\binom{n}{k}$ possible subsets.

In the next three sections, we present the corresponding solutions respectively.

Table 6.1: Symbols

Symbol	Definition and Description
$\mathbf{A}, \mathbf{B}, \dots$	matrices (bold upper case)
$\mathbf{A}(i, j)$	the element at the i^{th} row and j^{th} column of matrix \mathbf{A}
$\mathbf{A}(i, :)$	the i^{th} row of matrix \mathbf{A}
$\mathbf{A}(:, j)$	the j^{th} column of matrix \mathbf{A}
\mathbf{A}'	transpose of matrix \mathbf{A}
$\mathbf{a}, \mathbf{b}, \dots$	column vectors
$\mathcal{S}, \mathcal{T}, \dots$	sets (calligraphic)
n	number of nodes in the graph
m	number of edges in the graph
$(\lambda_i, \mathbf{u}_i)$	the i^{th} eigen-pair of \mathbf{A}
λ	first eigen-value of \mathbf{A} (i.e., $\lambda \triangleq \lambda_1$)
\mathbf{u}	first eigen-vector of \mathbf{A} (i.e., $\mathbf{u} \triangleq \mathbf{u}_1$)
$\lambda^{(i)}, \lambda^{(\mathcal{S})}$	first eigen-value of \mathbf{A} by deleting node i (or the set of nodes in \mathcal{S})
$\Delta\lambda(i)$	eigen-drop: $\Delta\lambda(i) = \lambda - \lambda^{(i)}$
$\Delta\lambda(\mathcal{S})$	eigen-drop: $\Delta\lambda(\mathcal{S}) = \lambda - \lambda^{(\mathcal{S})}$
$\text{Sv}(i)$	' <i>Shield-value</i> ' score of node i
$\text{Sv}(\mathcal{S})$	' <i>Shield-value</i> ' score of nodes in \mathcal{S}
$\text{V}(\mathbf{G})$	' <i>Vulnerability</i> ' score of the graph

6.4 Background: Our Solution for Problem 1

Here, we focus on Problem 6.1. We suggest using the first eigenvalue λ as the solution. We should point out that it *not* our main contribution to adopt λ as the '*Vulnerability*' measure of a graph. Nonetheless, it is the base of our proposed solutions for both Problem 2 and Problem 3.

6.4.1 '*Vulnerability*' Score

In Problem 6.1, the goal is to measure the '*Vulnerability*' of the whole graph by a single number. We adopt the first eigenvalue of the adjacency matrix \mathbf{A} as such a measurement (eq. (6.1)): the larger λ is, the more vulnerable the whole graph is.

$$\text{V}(\mathbf{G}) \triangleq \lambda \tag{6.1}$$

Figure 6.1 presents an example, where we have four graphs with 5 nodes. Intuitively, the graph becomes more and more vulnerable from the left to the right. In other words, for a given strength of the virus attack, it is more likely that an epidemic will break out

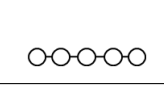
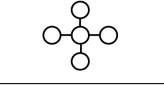
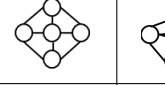
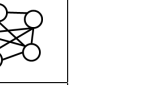
			
(a) $\lambda = 1.7$	(b) $\lambda = 2.0$	$\lambda = 2.9$	$\lambda = 4.0$

Figure 6.1: An example of measuring ‘Vulnerability’ of the graph. More edges, and carefully placed, make the graph better connected, and thus more vulnerable. Notice that the chain (a) and the star (b) have the same number of edges, but our λ score correctly considers the star as more vulnerable.

in the graphs on the right than those on the left side. Therefore, the vulnerability of the graph increases. We can see that the corresponding λ increases from left to right as well.

Notice that the concept of ‘Vulnerability’ is *different* from vertex connectivity of the graph [HT08]. For ‘Vulnerability’, we want to quantify how likely/easily a graph will be infected by a virus (given the strength of virus attack). Whereas for vertex connectivity, we want to quantify how difficult for a graph to be disconnected. For example, both graph (a) and (b) in figure 6.1 have the same vertex connectivity (both are 1s). But graph (b) is more vulnerable to the virus attack. Also notice that although ‘Vulnerability’ is related to both graph density (i.e., average degree) and diameter, neither of them can *fully* describe the ‘Vulnerability’ by itself. For example, in figure 6.1, (a) and (b) share the same density/average degree although (b) is more vulnerable than (a); (b) and (c) share the same diameter although (c) is more vulnerable than (b).

6.4.2 Justifications

The first eigenvalue λ is a good measurement of the graph ‘Vulnerability’, because of recent results on *epidemic thresholds* from immunology [CWW⁺08, PCF⁺11]: λ is closely related to the epidemic threshold τ of a graph under almost any epidemic model (including the flu-like SIS (susceptible-infective-susceptible) epidemic model), and specifically $\tau \propto 1/\lambda$. This means that a virus less infective than τ will quickly get extinguished instead of lingering forever. Therefore, given the strength of the virus (that is, the infection rate and the host-recovery rate), an epidemic is more likely for a graph with larger λ .

We can also show that the first eigenvalue λ is closely related to the so-called *loop capacity* and the *path capacity* of the graph, that is, the number of loops and paths of length l ($l = 2, 3, \dots$). If a graph has many such loops and paths, then it is well connected, and thus more vulnerable (i.e., it is easier for a virus to propagate across the graph = the graph is less robust to the virus attack).

6.5 Our Solution for Problem 2

In this section, we focus on Problem 6.4. We first present our solution, and then provide justifications.

6.5.1 Proposed ‘Shield-value’ Score

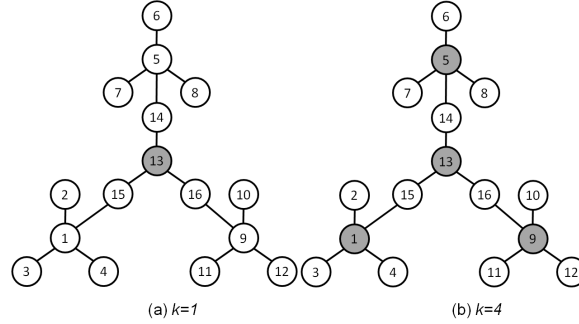


Figure 6.2: An example on measuring the ‘Shield-value’ score of a given set of nodes. The best k nodes found by our NETSHIELD are shaded. In (a), notice that the highest degree nodes (e.g., node 1) is not chosen. In (b), immunizing the shaded nodes makes the remaining graph most robust to the virus attack.

In Problem 6.4, the goal is to quantify the importance of a given set of nodes \mathcal{S} , and specifically the impact of their deletion/immunization to the ‘Vulnerability’ of the rest of the graph. The obvious choice is the drop in eigenvalue, or *eigen-drop* $\Delta\lambda$ that their removal will cause to the graph. We propose to approximate it, to obtain efficient computations, as we describe later. Specifically, we propose using $Sv(\mathcal{S})$ defined as:

$$Sv(\mathcal{S}) = \sum_{i \in \mathcal{S}} 2\lambda \mathbf{u}(i)^2 - \sum_{i, j \in \mathcal{S}} \mathbf{A}(i, j) \mathbf{u}(i) \mathbf{u}(j) \quad (6.2)$$

Intuitively, by eq. (6.2), a set of nodes \mathcal{S} has higher ‘Shield-value’ score if (1) each of them has a high eigen-score ($\mathbf{u}(i)$), and (2) they are dissimilar with each other (small or zero $\mathbf{A}(i, j)$). Figure 6.2 shows an example on measuring the ‘Shield-value’ score of a given set of nodes. The best k nodes found by our NETSHIELD (which will be introduced very soon in the next section) are shaded. The result is consistent with intuition. In figure 6.2(a), it picks node 13 as best $k = 1$ node (although nodes 1, 5 and 9 have the highest degree). In figure 6.2(b), deleting the shaded nodes (node 1, 5, 9 and 13) will make the graph the least vulnerable (i.e., the remaining graphs are sets of isolated nodes; and therefore it is most robust to virus attack).

6.5.2 Justifications

Here, we provide some justifications on the proposed ‘Shield-value’ score, which is summarized in Lemma 6.1. It says that our proposed ‘Shield-value’ score $Sv(\mathcal{S})$ is a good approximation for the eigen-drop $\Delta\lambda(\mathcal{S})$ when deleting the set of nodes \mathcal{S} from the original graph \mathbf{A} .

Lemma 6.1. Let $\lambda^{(\mathcal{S})}$ be the (exact) first eigen-value of $\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$ is the perturbed version of \mathbf{A} by removing all of its rows/columns indexed by set \mathcal{S} . Let $\delta = \lambda - \lambda_2$ be the eigen-gap, and d be the maximum degree of \mathbf{A} . If λ is the simple first eigen-value of \mathbf{A} , and $\delta \geq 2\sqrt{2kd}$, then

$$\Delta\lambda(\mathcal{S}) = Sv(\mathcal{S}) + O\left(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2\right) \quad (6.3)$$

where $Sv(\mathcal{S})$ is computed by eq. (6.2) and $\Delta\lambda(\mathcal{S}) = \lambda - \lambda^{(\mathcal{S})}$.

Proof: First, let us write $\hat{\mathbf{A}}$ as a perturbed version of the original matrix \mathbf{A} :

$$\hat{\mathbf{A}} = \mathbf{A} + \mathbf{E}, \quad \text{and} \quad \mathbf{E} = \mathbf{F} + \mathbf{F}' + \mathbf{G} \quad (6.4)$$

where $\mathbf{F}(:,j) = -\mathbf{A}(:,j)$ ($j \in \mathcal{S}$ and $\mathbf{F}(:,j) = 0$ ($j \notin \mathcal{S}$); $\mathbf{G}(i,j) = \mathbf{A}(i,j)$ ($i,j \in \mathcal{S}$) and $\mathbf{G}(i,j) = 0$ ($i \notin \mathcal{S}$, or $j \notin \mathcal{S}$).

Since $\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$, we have

$$\begin{aligned} \mathbf{u}'\mathbf{F}'\mathbf{u} &= \mathbf{u}'\mathbf{F}\mathbf{u} = (\mathbf{F}'\mathbf{u})'\mathbf{u} = -\sum_{j \in \mathcal{S}} \lambda\mathbf{u}(j)^2 \\ \mathbf{u}'\mathbf{G}\mathbf{u} &= \sum_{i,j \in \mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \end{aligned} \quad (6.5)$$

Let $\tilde{\lambda}$ be the corresponding perturbed eigen-value of λ , according to the matrix perturbation theory (p.183 [SS90]), we have

$$\begin{aligned} \tilde{\lambda} &= \lambda + \mathbf{u}'\mathbf{E}\mathbf{u} + O(\|\mathbf{E}\|^2) \\ &= \lambda + \mathbf{u}'\mathbf{F}\mathbf{u} + \mathbf{u}'\mathbf{F}'\mathbf{u} + \mathbf{u}'\mathbf{G}\mathbf{u} + O(\|\mathbf{E}\|^2) \\ &= \lambda - \left(\sum_{j \in \mathcal{S}} 2\lambda\mathbf{u}(j) - \sum_{i,j \in \mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)\right) \\ &\quad + O\left(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2\right) \\ &= \lambda - Sv(\mathcal{S}) + O\left(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2\right) \end{aligned} \quad (6.6)$$

Let $\tilde{\lambda}_i$ ($i = 2, \dots, n$) be the corresponding perturbed eigen-value of λ_i ($i = 2, \dots, n$). Again, by the matrix perturbation theory (p.203 [SS90]), we have

$$\begin{aligned} \tilde{\lambda} &\geq \lambda - \|\mathbf{E}\|_2 \geq \lambda - \|\mathbf{E}\|_F \geq \lambda - \sqrt{2kd} \\ \tilde{\lambda}_i &\leq \lambda_i + \|\mathbf{E}\|_2 \leq \lambda_i + \|\mathbf{E}\|_F \leq \lambda_i + \sqrt{2kd} \end{aligned} \quad (6.7)$$

Since $\delta = \lambda - \lambda_2 \geq 2\sqrt{2kd}$, we have $\tilde{\lambda} \geq \tilde{\lambda}_i$ ($i = 2, \dots, n$). In other words, we have $\lambda^{(\mathcal{S})} = \tilde{\lambda}$. Therefore,

$$\begin{aligned} \Delta\lambda(\mathcal{S}) &= \lambda - \lambda^{(\mathcal{S})} = \lambda - \tilde{\lambda} \\ &= Sv(\mathcal{S}) + O\left(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2\right) \end{aligned} \quad (6.8)$$

which completes the proof. \square

6.6 Our Solution for Problem 3

In this section, we deal with Problem 6.3. Here, the goal is to find a subset of k nodes with the highest ‘*Shield-value*’ score (among all $\binom{n}{k}$ possible subsets). We start by showing that the two straight-forward methods (referred to as ‘Com-Eigs’, and ‘Com-Eval’) are computationally intractable. Then, we present the proposed NETSHIELD algorithm. Finally, we analyze its accuracy as well as its computational complexity.

6.6.1 Preliminaries

There are two obviously straight-forward methods for Problem 6.3. The first one (referred to as ‘Com-Eigs’¹) works as follows: for each possible subset \mathcal{S} , we delete the corresponding rows/columns from the adjacency matrix \mathbf{A} ; compute the first eigenvalue of the new perturbed adjacency matrix; and finally output the subset of nodes which has the smallest eigenvalue (therefore has the largest eigen-drop). Despite the simplicity of this strategy, it is computational intractable due to its combinatorial nature. It is easy to show that the computational complexity of ‘Com-Eigs’ is $O(\binom{n}{k} \cdot m)^2$. This is computationally intractable even for small graphs. For example, in a graph with 1K nodes and 10K edges, suppose that it takes about 0.01 second to find its first eigen-value. Then we need about 2,615 years to find the best-5 nodes with the highest ‘*Shield-value*’ score!

A more reasonable (in terms of speed) way to find the best- k nodes is to evaluate $Sv(\mathcal{S})$, rather than to compute the first eigen-value $\lambda_{\mathcal{S}}$, $\binom{n}{k}$ times, and pick the subset with the highest $Sv(\mathcal{S})$. We refer to this strategy as ‘Com-Eval’. Compared with the straight-forward method (referred to as ‘Com-Eigs’, which is $O(\binom{n}{k} \cdot m)$); ‘Com-Eval’ is much faster ($O(\binom{n}{k} \cdot k^2)$). However, ‘Com-Eval’ is still not applicable to real applications due to its combinatorial nature. Again, in a graph with 1K nodes and 10K edges, suppose that it only takes about 0.00001 second to evaluate $Sv(\mathcal{S})$ once. Then we still need about 3 months to find the best-5 nodes with the highest ‘*Shield-value*’ score!

6.6.2 Proposed NETSHIELD Algorithm

The proposed NETSHIELD is given in Alg. 1. In Alg. 1, we compute the first eigenvalue λ and the corresponding eigenvector \mathbf{u} in step 1. In step 4, the $n \times 1$ vector \mathbf{v} measures the ‘*Shield-value*’ score of each individual node. Then, in each iteration of steps 6-17, we greedily select one more node and add it into set \mathcal{S} according to $\text{score}(j)$ (step 13). Note that steps 10-12 are to exclude those nodes that are already in the selected set \mathcal{S} .

¹In fact, we can prove Problem 6.3 is NP-hard, using a slight variant of the proof of Theorem [fill] in Chapter 7.

²We assume that k is relatively small compared with n and m (e.g., tens or hundreds). Therefore, after deleting k rows/columns from \mathbf{A} , we still have $O(m)$ edges.

Algorithm 1 NETSHIELD

Input: the adjacency matrix \mathbf{A} and an integer k

Output: a set \mathcal{S} with k nodes

```
1: compute the first eigen-value  $\lambda$  of  $\mathbf{A}$ ; let  $\mathbf{u}$  be the corresponding eigen-vector  $\mathbf{u}(j)$  ( $j = 1, \dots, n$ );
2: initialize  $\mathcal{S}$  to be empty;
3: for  $j = 1$  to  $n$  do
4:    $\mathbf{v}(j) = (2 \cdot \lambda - \mathbf{A}(j, j)) \cdot \mathbf{u}(j)^2$ ;
5: end for
6: for  $\text{iter} = 1$  to  $k$  do
7:   let  $\mathbf{B} = \mathbf{A}(:, \mathcal{S})$ ;
8:   let  $\mathbf{b} = \mathbf{B} \cdot \mathbf{u}(\mathcal{S})$ ;
9:   for  $j = 1$  to  $n$  do
10:    if  $j \in \mathcal{S}$  then
11:      let  $\text{score}(j) = -1$ ;
12:    else
13:      let  $\text{score}(j) = \mathbf{v}(j) - 2 \cdot \mathbf{b}(j) \cdot \mathbf{u}(j)$ ;
14:    end if
15:  end for
16:  let  $i = \text{argmax}_j \text{score}(j)$ , add  $i$  to set  $\mathcal{S}$ ;
17: end for
18: return  $\mathcal{S}$ .
```

6.6.3 Analysis of NETSHIELD

Here, we analyze the accuracy and efficiency of the proposed NETSHIELD.

First, according to the following theorem, Alg. 1 is *near-optimal* wrt ‘Com-Eval’. In addition, by Lemma 6.1, our ‘Shield-value’ score (which ‘Com-Eval’ tries to optimize) is a good approximation for the actual eigen-drop $\Delta\lambda(\mathcal{S})$ (which ‘Com-Eigs’ tries to optimize). Therefore, we would expect that Alg. 1 also gives a good approximation wrt ‘Com-Eigs’ (See Section 6.7 for experimental validation).

Theorem 6.1. Effectiveness of NETSHIELD. *Let \mathcal{S} and $\tilde{\mathcal{S}}$ be the sets selected by Alg. 1 and by ‘Com-Eval’, respectively. Let $\Delta\lambda(\mathcal{S})$ and $\Delta\lambda(\tilde{\mathcal{S}})$ be the corresponding eigen-drops. Then, $\Delta\lambda(\mathcal{S}) \geq (1 - 1/e)\Delta\lambda(\tilde{\mathcal{S}})$.*

Proof: Let $\mathcal{J}, \mathcal{J}, \mathcal{K}$ be three sets and $\mathcal{J} \subseteq \mathcal{J}$. Define the following three sets based on $\mathcal{J}, \mathcal{J}, \mathcal{K}$: $\mathcal{S} = \mathcal{J} \cup \mathcal{K}$, $\mathcal{T} = \mathcal{J} \cup \mathcal{K}$, $\mathcal{R} = \mathcal{J} \setminus \mathcal{J}$.

Substituting eq.(6.2), we have

$$\begin{aligned}
Sv(\mathcal{S}) - Sv(\mathcal{J}) &= \sum_{i \in \mathcal{K}} 2\lambda \mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \\
&\quad - 2 \sum_{j \in \mathcal{J}, i \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \\
Sv(\mathcal{T}) - Sv(\mathcal{J}) &= \sum_{i \in \mathcal{K}} 2\lambda \mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \\
&\quad - 2 \sum_{j \in \mathcal{J}, i \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j)
\end{aligned} \tag{6.9}$$

According to Perron-Frobenius theorem, we have $\mathbf{u}(i) \geq 0 (i = 1, \dots, n)$. Therefore,

$$\begin{aligned}
(Sv(\mathcal{S}) - Sv(\mathcal{J})) - (Sv(\mathcal{T}) - Sv(\mathcal{J})) \\
&= 2 \sum_{i \in \mathcal{K}, j \in \mathcal{R}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \geq 0 \\
&\Rightarrow Sv(\mathcal{S}) - Sv(\mathcal{J}) \geq Sv(\mathcal{T}) - Sv(\mathcal{J})
\end{aligned} \tag{6.10}$$

Therefore, the function $Sv(\mathcal{S})$ is sub-modular.

Next, we can verify that node i selected in step 16 of Alg. 1 satisfies $i = \operatorname{argmax}_{j \notin \mathcal{S}} Sv(\mathcal{S} \cup j)$ for a fixed set \mathcal{S} .

Next, we prove that $Sv(\mathcal{S})$ is monotonically non-decreasing wrt \mathcal{S} . According to eq. (6.9), we have

$$\begin{aligned}
Sv(\mathcal{S}) - Sv(\mathcal{J}) &= \sum_{i \in \mathcal{K}} 2\lambda \mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \\
&\quad - 2 \sum_{j \in \mathcal{J}, i \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \\
&\geq \sum_{i \in \mathcal{K}} 2\lambda \mathbf{u}(i)^2 - 2 \sum_{j \in \mathcal{S}, i \in \mathcal{K}} \mathbf{A}(i,j) \mathbf{u}(i) \mathbf{u}(j) \\
&= 2 \sum_{i \in \mathcal{K}} \mathbf{u}(i) (\lambda \mathbf{u}(i) - \sum_{j \in \mathcal{S}} \mathbf{A}(i,j) \mathbf{u}(j)) \\
&\geq 2 \sum_{i \in \mathcal{K}} \mathbf{u}(i) (\lambda \mathbf{u}(i) - \sum_{j=1}^n \mathbf{A}(i,j) \mathbf{u}(j)) \\
&= 2 \sum_{i \in \mathcal{K}} \mathbf{u}(i) (\lambda \mathbf{u}(i) - \lambda \mathbf{u}(i)) = 0
\end{aligned} \tag{6.11}$$

where the second last equality is due to the definition of eigenvalue.

Finally, it is easy to verify that $Sv(\emptyset) = 0$, where \emptyset is an empty set. Using the property of sub-modular functions [KG07], we have $\Delta\lambda(\mathcal{S}) \geq (1 - 1/e)\Delta\lambda(\hat{\mathcal{S}})$. \square

According to Lemma 6.2, the computational complexity of Alg. 1 is $O(nk^2 + m)$, which is much faster than both ‘Com-Eigs’ ($O(\binom{n}{k}m)$) and ‘Com-Eval’ ($O(\binom{n}{k}k^2)$).

Lemma 6.2. Computational Complexity of NETSHIELD. *The computational complexity of Alg. 1 is $O(nk^2 + m)$.*

Proof: Omitted for brevity. □

Finally, according to Lemma 6.3, the space cost of Alg. 1 is also efficient (i.e., linear wrt the size of the graph).

Lemma 6.3. Space Cost of NETSHIELD. *The space cost of Alg. 1 is $O(n + m + k)$.*

Proof: Omitted for brevity. □

6.7 Experimental Evaluations (Static Graphs)

We present detailed experimental results in this section. All the experiments are designed to answer the following questions:

- 1: (*Effectiveness*) How effective is the proposed $Sv(S)$ in real graphs?
- 2: (*Efficiency*) How fast and scalable is the proposed NETSHIELD?

6.7.1 Data sets

Table 6.2: Summary of the data sets

Name	n	m
<i>Karate</i>	34	152
<i>AA</i>	418,236	2,753,798
<i>NetFlix</i>	2,667,199	171,460,874

We used three real data sets, which are summarized in table 6.2. The first data set (*Karate*) is a unipartite graph, which describes the friendship among the 34 members of a karate club at a US university [Zac77]. Each node is a member in the karate club and the existence of the edge indicates that the two corresponding members are friends. Overall, we have $n = 34$ nodes and $m = 156$ edges.

The second data set (*AA*) is an author-author network from DBLP.³ *AA* is a co-authorship network, where each node is an author and the existence of an edge indicates the co-authorship between the two corresponding persons. Overall, we have $n =$

³<http://www.informatik.uni-trier.de/~ley/db/>

418,236 nodes and $m = 2,753,798$ edges. We also construct much smaller co-authorship networks, using the authors from only one conference (e.g., *KDD*, *SIGIR*, *SIGMOD*, etc.). For example, *KDD* is the co-authorship network for the authors in the ‘KDD’ conference. For these smaller co-authorship networks, they typically have a few thousand nodes and up to a few ten thousand edges.

The last data set (*NetFlix*) is from the Netflix prize.⁴ This is also a bipartite graph. We have two types of nodes: user and movie. The existence of an edge indicates that the corresponding user has rated the corresponding movie. Overall, we have $n = 2,667,199$ nodes and $m = 171,460,874$ edges. This is a bipartite graph, and we convert it to a unipartite graph \mathbf{A} : $\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}' & \mathbf{0} \end{pmatrix}$, where $\mathbf{0}$ is a matrix with all zero entries and \mathbf{B} is the adjacency matrix of the bipartite graph.

6.7.2 Effectiveness

Here, we first test the approximation accuracy of the proposed $Sv(\mathcal{S})$. Then, we compared the different immunization policies, followed by some case studies. Notice that the quality vs. speed trade-off for the proposed NETSHIELD, the optimal ‘Com-Eigs’ and the alternative greedy method is presented in subsection 6.7.3.

Approximation quality of $Sv(\mathcal{S})$

The proposed NETSHIELD is based on eq. (6.2). That is, we want to approximate the first eigen-value of the perturbed matrix by λ and \mathbf{u} . By Lemma 6.1, it says that $Sv(\mathcal{S})$ is a good approximation for the actual eigen-drop $\Delta\lambda(\mathcal{S})$. Here, let us experimentally evaluate how good this approximation is on real graphs. We construct an authorship network from one of the following conferences: ‘*KDD*’, ‘*ICDM*’, ‘*SDM*’ and ‘*SIGMOD*’. We then compute the linear correlation coefficient between $\Delta\lambda(\mathcal{S})$ and $Sv(\mathcal{S})$ with several different k values ($k = 1, 2, 5, 10, 20$). The results are shown in table 6.3. It can be seen that the approximation is very good - in all the cases, the linear correlation coefficient is greater than 0.95. Figure 6.3 gives the scatter plot of $\Delta\lambda(\mathcal{S})$ (i.e., the actual eigen-drop) vs. $Sv(\mathcal{S})$ (i.e., the proposed ‘*Shield-value*’) for $k = 5$ the on ‘*ICDM*’ data set.

Immunization by NETSHIELD

The proposed ‘*Vulnerability*’ score of the graph is motivated by the epidemic threshold [PCF⁺11]. As a consequence, the proposed NETSHIELD leads to a natural immunization strategy for the SIS model (susceptible-infective-susceptible, like, e.g., the flu): quarantine or delete the subset of the nodes detected by NETSHIELD in order to prevent

⁴<http://www.netflixprize.com/>

Table 6.3: Evaluation on the approximation accuracy of $f(S)$. Larger is better.

k	'KDD'	'ICDM'	'SDM'	'SIGMOD'
1	0.9519	0.9908	0.9995	1.0000
2	0.9629	0.9910	0.9984	0.9927
5	0.9721	0.9888	0.9992	0.9895
10	0.9726	0.9863	0.9987	0.9852
20	0.9683	0.9798	0.9929	0.9772

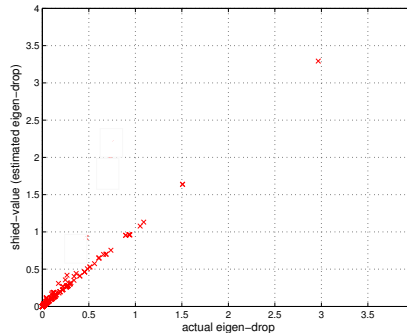


Figure 6.3: Evaluation of the approximation accuracy of $S_v(S)$ on the 'ICDM' graph. The proposed 'Shield-value' S_v (y -axis) gives a good approximation for the actual eigen-drop $\Delta\lambda(S)$ (x -axis). Most points are on or close to the diagonal (ideal).

an epidemic from breaking out.⁵

We compare it with the following alternative choices: (1) picking a random neighbor of a randomly chosen node [CHbA03] ('Acquaintance'), (2) picking the nodes with the highest eigen scores $\mathbf{u}(i)$ ($i = 1, \dots, n$) ('Eigs')⁶, (3) picking the nodes with the highest abnormality scores [SQCF05] ('abnormality'), (4) picking the nodes with the highest betweenness centrality scores based on the shortest path [Fre77] ('Short'), (5) picking the nodes with the highest betweenness centrality scores based on random walks [New05b] ('N.RW'), (6) picking the nodes with the highest degree ('Degree'), and (7) picking the nodes with the highest PageRank scores [PBMW98] ('PageRank'). For each method, we delete 5 nodes for immunization. Let $s = \lambda \cdot b/d$ be the normalized virus strength (bigger s means more stronger virus), where b and d are the infection rate and death rate, respectively. The result is presented in figure 6.4, which is averaged over 1000 runs. It can be seen that the proposed NETSHIELD is always the best, - its curve is always the lowest which means that we always have the least number of infected nodes in the graph with this immunization strategy. Notice that the performance of

⁵In fact our NETSHIELD will also help with the immunization for almost any model (as the threshold is dependent on λ).

⁶For the un-directed graph which we focus on in this work, 'Eigs' is equivalent to 'HITS' [Kle98].

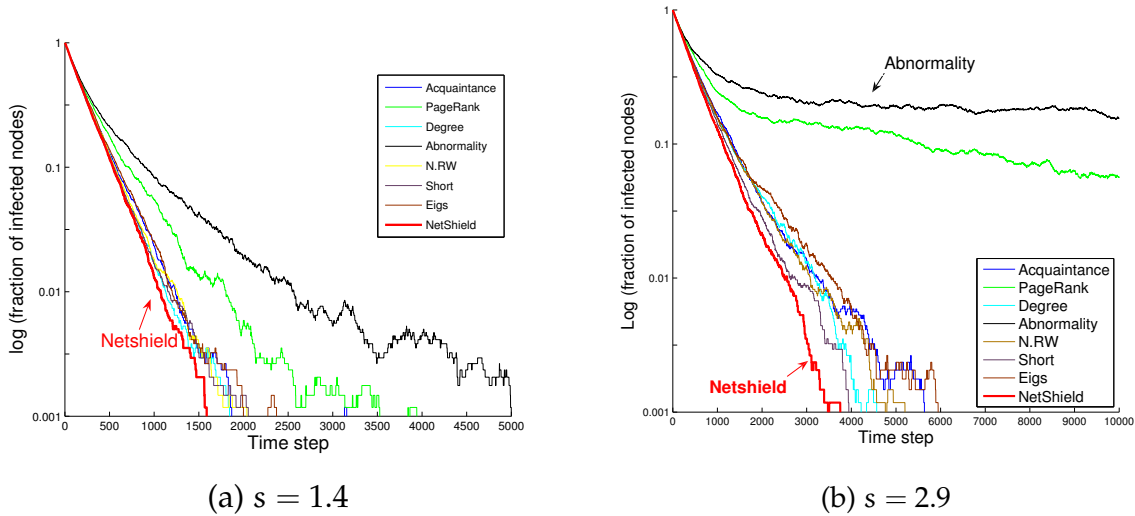


Figure 6.4: Evaluation of immunization of NETSHIELD on the Karate graph. The fraction of infected nodes (in log-scale) vs. the time step. s is normalized virus strength. Lower is better. The proposed NETSHIELD is always the best, leading to the fastest healing of the graph. Best viewed in color.

‘Eigs’ is much worse than the proposed NETSHIELD. This indicates that by *collectively* finding a set of nodes with the highest ‘Shield-value’, we indeed obtain extra performance gain (compared with naively choosing the top- k nodes which have the highest *individual* ‘Shield-value’ scores).

Case studies

Next, we will show some case studies to illustrate the effectiveness of the proposed $Sv(S)$ as a ‘Shield-value’ score of a subset of nodes.

We run the proposed NETSHIELD on AA data set and return the best $k = 200$ authors. Some representative authors, to name a few, are ‘Sudhakar M. Reddy’ ‘Wei Wang’ ‘Heinrich Niemann’, ‘Srimat T. Chakradhar’, ‘Philip S. Yu’, ‘Lei Zhang’, ‘Wei Li’, ‘Jiawei Han’, ‘Srinivasan Parthasarathy’, ‘Srivaths Ravi’, ‘Antonis M. Paschalis’, ‘Mohammed Javeed Zaki’, ‘Lei Li’, ‘Dimitris Gizopoulos’, ‘Alberto L. Sangiovanni-Vincentelli’, ‘Narayanan Vijaykrishnan’, ‘Jason Cong’, ‘Thomas S. Huang’, etc. We can make some very interesting observations from the result:

- 1 There are some multi-disciplinary people in the result. For example, Prof. Alberto L. Sangiovanni-Vincentelli from UC Berkeley is interested in ‘design technology’, ‘cad’, ‘embedded systems’, and ‘formal verification’; Prof. Philip S. Yu from UIC is interested in ‘databases’, ‘performance’, ‘distributed systems’ and ‘data mining’.
- 2 Some people show up because they are famous in one specific area, and occasionally have one/two papers in a remotely related area (therefore, increasing the path capacity between two remote areas). For example, Dr. Srimat T. Chakradhar mainly

focuses on ‘cad’. But he has co-authored in a ‘NIPS’ paper. Therefore, he creates a critical connection between these two (originally) remote areas: ‘cad’ and ‘machine learning’.

- 3 Some people show up because they have ambiguous names (e.g., Wei Wang, Lei Li, Lei Zhang, Wei Li, etc.). Take ‘Wei Wang’ as an example; according to DBLP,⁷ there are 7 different ‘Wei Wang’s. In our experiment, we treat all of them as one person. That is to say, it is equivalent to putting an artificial ‘Wei Wang’ in the graph who brings 7 different ‘Wei Wang’s together. These 7 ‘Wei Wang’s are in fact spread out in quite different areas. (e.g., Wei Wang@UNC is in ‘data mining’ and ‘bio’; Wei Wang@NUS is in ‘communication’; Wei Wang@MIT is in ‘non-linear systems’.)

6.7.3 Efficiency

We will study the wall-clock running time of the proposed NETSHIELD here. Basically, we want to answer the following three questions:

1. (*Speed*) What is the speedup of the proposed NETSHIELD over the straight-forward methods (‘Com-Eigs’ and ‘Com-Eval’)?
2. (*Scalability*) How does NETSHIELD scale with the size of the graph (n and m) and k ?
3. (*Quality/Speed Trade-Off*) How does NETSHIELD balance between the quality and the speed?

For the results we report in this subsection, all of the experiments are done on the same machine with four 2.4GHz AMD CPUs and 48GB memory, running Linux (2.6 kernel). If the program takes more than 1,000,000 seconds, we stop running it.

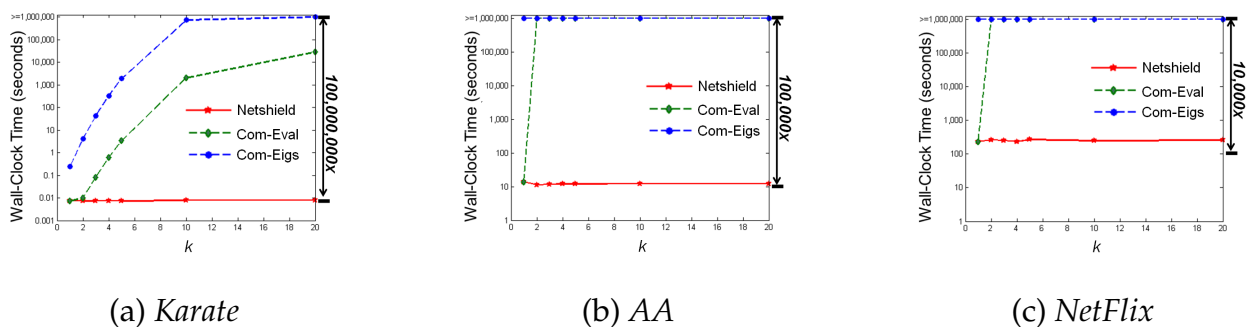
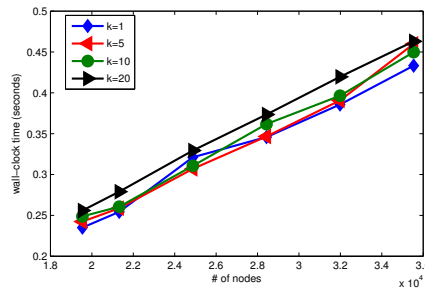


Figure 6.5: Wall-clock time vs. the budget k for different methods. The time is in the logarithmic scale. Our NETSHIELD (red star) is much faster. Lower is better.

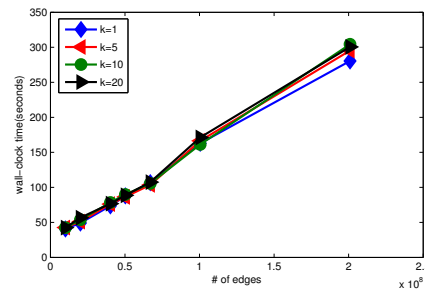
⁷<http://www.informatik.uni-trier.de/~ey/db/indices/a-tree/w/Wang:Wei.html>

First, we compare NETSHIELD with ‘Com-Eigs’ and ‘Com-Eval’. Figure 6.5 shows the comparison on three real data sets. We can make the following conclusions: (1) Straight-forward methods (‘Com-Eigs’ and ‘Com-Eval’) are computationally intractable even for a small graph. For example, on the *Karate* data set with only 34 nodes, it takes more than 100,000 and 1,000 seconds to find the best-10 by ‘Com-Eigs’ and by ‘Com-Eval’, respectively. (2) The speedup of the proposed NETSHIELD over both ‘Com-Eigs’ and ‘Com-Eval’ is huge - in most cases, we achieve *several (up to 7) orders of magnitude* speedups! (3) The speedup of the proposed NETSHIELD over both ‘Com-Eigs’ and ‘Com-Eval’ quickly increases wrt the size of the graph as well as k . (4) For a given size of the graph (fixed n and m), the wall-clock time is almost constant - suggesting that NETSHIELD spends most of its running time in computing λ and \mathbf{u} .

Next, we evaluate the scalability of NETSHIELD. From figure 6.6, it can be seen that NETSHIELD scales linearly wrt both n and m , which means that it is suitable for large graphs.



(a) changing n (fix $m = 119,460$)



(b) changing m (fix $n = 2,667,119$)

Figure 6.6: Evaluation of the scalability of the proposed NETSHIELD wrt. n (number of nodes) and m (number of edges), respectively. The wall-clock time of our NETSHIELD scales linearly wrt n and m .

Finally, we evaluate how the proposed NETSHIELD balances between the quality and speed. For the *Karate* graph, we use the proposed NETSHIELD to find a set of k nodes and check the corresponding eigen-drop (i.e., the decrease of the first eigen-value of the adjacency matrix) as well as the corresponding wall-clock time. We compare it

with ‘Com-Eigs’, which always gives the optimal solutions (i.e., it returns the subset that leads to the largest eigen-drop). The results (eigen-drop vs. wall-clock time) are plotted in figure 6.7. It can be seen that NETSHIELD gains significant of speedup over the ‘Com-Eigs’, at the cost of a small fraction of quality loss (i.e., the green dash lines are near-flat).

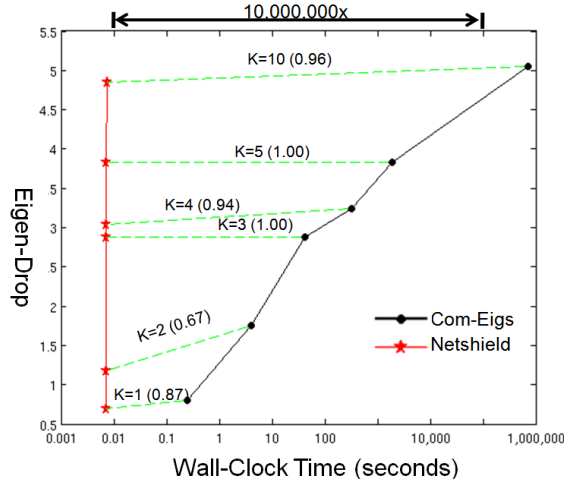


Figure 6.7: Evaluation of the quality/speed trade off. Eigen-drop vs.wall-clock time, with different budget k .The proposed NETSHIELD (red star) achieves a good balance between eigen-drop and speed. Note that the x-axis (wall-clock time) is in logarithmic scale. The number inside the parenthesis above each green dash curve is the ratio of eigen-drop between NETSHIELD and ‘Com-Eigs’. NETSHIELD is optimal when this ratio is 1. Best viewed in color.

We also compare the proposed NETSHIELD with the following heuristic (referred to as ‘Greedy’): at each iteration, we re-compute the first eigenvector of the *current* graph and pick a node with the highest eigen-score $u(i)$; then we delete this node from the graph and go to the next iteration. For the *Netflix* graph, we find a set of k nodes and check the corresponding eigen-drop as well as the corresponding wall-clock time. The quality /speed trade-off curve is plotted in figure 6.8. From the figure, we can make two observations: (1) the quality of the two methods (‘Greedy’ vs. the proposed NETSHIELD) are almost the same (note that the green dash curves in the plots are always straight flat); (2) the proposed NETSHIELD is always faster than ‘Greedy’ (up to 103x speedup).

6.8 Immunization under time-varying graphs

We now tackle the problem of immunization under a time-varying network. Consider a setting with clearly different behaviors say, day/night, each characterized by a corresponding adjacency matrix (A_1 for day, A_2 for night)—what are the best nodes to immunize to prevent an epidemic as much as possible? We restrict our attention to the

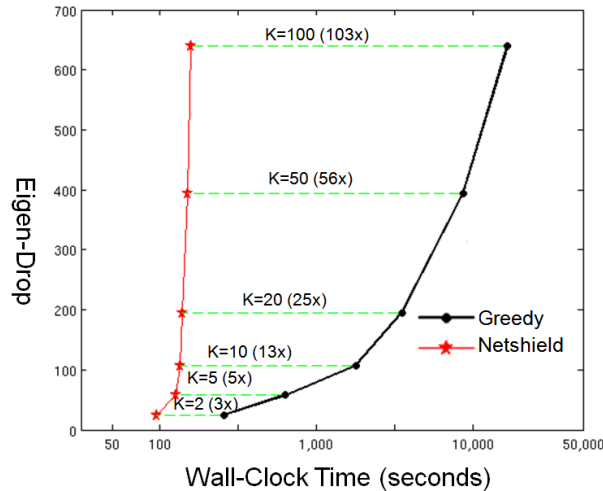


Figure 6.8: Comparison of NETSHIELD vs. ‘Greedy’. The proposed NETSHIELD (red star) is better than ‘Greedy’ (i.e., faster, with the same quality). Note that the x-axis (wall-clock time) is in logarithmic scale. The number inside the parenthesis above each green dash curve is the speedup of the proposed NETSHIELD over ‘Greedy’. Best viewed in color.

SIS-model only for this problem. More generally, the problem we are tackling can be formally stated as follows:

Problem 6.4. Dynamic Immunization

Given: (1) T alternating behaviors, characterized by a set of T graphs $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_2 \dots, \mathbf{A}_T\}$; and (2) the SIS model with virus parameters β and δ and (3) k vaccines;

Find: The best- k nodes for immunization.

6.8.1 Quality Metric

Using our results in Chapter 3, we can evaluate the quality of *any* immunization policy. Recall that there is no epidemic if the largest eigenvalue of the corresponding ‘system matrix’ is below one in magnitude. Hence, smaller the value of λ_S , lesser are chances of the virus causing any epidemic. Put differently, while immunizing, we want to decrease the λ_S value of the system as much as possible. Thus, the efficacy of any immunization policy should be measured using the amount of “drop” in λ_S it causes and the resulting λ_S after immunization.

6.8.2 Proposed immunization policies

We now discuss some new immunization policies for time-varying graphs, partially motivated by known policies used for static graphs. Again, for ease of exposition we focus our attention only on the $\{\mathbf{A}_1, \mathbf{A}_2\}$ system of Section 3.3. From the above, it is clear

that optimally we should choose that set of k nodes which result in the smallest λ_S value possible after immunization. This implies that for each set of k node we test, we need to delete k rows/columns from both \mathbf{A}_1 and \mathbf{A}_2 to get new matrices \mathbf{A}_1^* and \mathbf{A}_2^* and then compute the new λ_S value. The number of k sets is $\binom{n}{k}$ and therefore this method is combinatorial in nature and will be intractable even for small graphs. Nevertheless, we call this strategy `Optimal` and show experimental results for this policy too, because this policy will give us the lower-bound on the λ_S that can be achieved after k immunizations.

We want policies which are practical for large graphs and at the same time be efficient in lowering the λ_S value of the system i.e. they should be close to `Optimal`. Specifically to this effect, we now present several greedy policies as well. As the heuristics are greedy in nature, we only describe how to pick the best *one* node for immunization from a given set of G_1 and G_2 graphs. Our proposed policies are:

Greedy-DmaxA (Highest degree on \mathbf{A}_1 or \mathbf{A}_2 matrices) Under this policy, at each step we select the node with the highest degree considering both the \mathbf{A}_1 or \mathbf{A}_2 adjacency matrices. This is motivated by the degree immunization strategy used for static graphs.

Greedy-DavgA (Highest degree on the “average” matrix) We select the node with the highest degree in the \mathbf{A}_{avg} matrix where $\mathbf{A}_{avg} = (\mathbf{A}_1 + \mathbf{A}_2)/2$.

Greedy-AavgA (Acquaintance immunization [CHbA03] on the average matrix) The “acquaintance” immunization policy works by picking a random person, and then immunizing one of its neighbors at random (which will probably be a ‘hub’). We run this policy on the \mathbf{A}_{avg} matrix.

Greedy-S (Greedy on the system-matrix) This is the greedy strategy of immunizing the node at each step which causes the largest drop in λ_S value. Note that even this can be expensive for large graphs as we have to evaluate the first eigenvalue of \mathbf{S} after deleting each node to decide which node is the best.

Optimal Finally, this is the optimal through combinatorial strategy mentioned above of finding the best- k set of nodes which decrease λ_S the most.

6.8.3 Experimental Setup

We conducted a series of experiments using the MIT Reality Mining data set [EPL09]. The Reality Mining data consists of 104 mobile devices (cellular phones) monitored over a period of nine months (September 2004 - June 2005). If another participating Bluetooth device was within a range of approximately 5-10 meters, the date and time of the contact and the device’s MAC address were recorded. Bluetooth scans were conducted at 5-minute intervals and aggregated into two 12-hour period adjacency matrices (*day* and *night*). The epidemic simulations were accomplished by alternating the *day* and *night* matrices over the period of simulation. All experiments were run on a 64-bit, quad-core (2.5Ghz each) server running a CentOS linux distribution with shared 72 GB of RAM. Simulations were conducted using a combination of Matlab 7.9 and Python 2.6.

6.8.4 Results

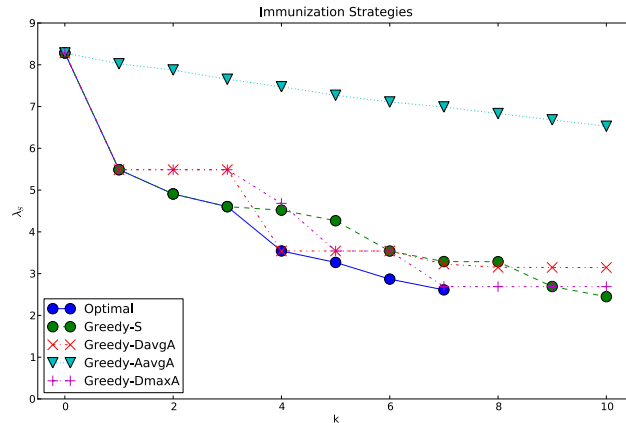


Figure 6.9: Experiments on Reality Mining graphs: λ_S vs k for different immunization policies. Lower is better. Greedy-DavgA clearly drops the λ_S value aggressively and is close to the Greedy-Opt.

Figure 6.9 shows the λ_S value after immunizing $k = 1, 2, \dots, 10$ nodes using each of the policies outlined above. As Optimal and Greedy-S require β and δ as inputs, we set $\beta = 0.5, \delta = 0.1$. Running Optimal became prohibitively expensive (> 4 hours on the MIT reality graphs) after $k = 7$ - hence we don't show data points for $k \geq 8$ for Optimal. Moving on to the greedy strategies we find that Greedy-S performs the best after $k = 10$ by dropping the λ_S value as aggressively as possible - equal to Optimal at many places. We find that Greedy-DavgA also performs very well. Intuitively this is because the highest degree node in \mathbf{A}_{avg} is very well-connected and hence has a huge effect in reducing the \mathbf{A}_{avg} value (we discuss more about \mathbf{A}_{avg} below later). At the same time, Greedy-DmaxA is comparable to Greedy-DavgA as we find the highest degree among both the graphs: so this highest degree will also mostly have the highest mean degree. Finally, Greedy-AavgA drops the λ_S value the least among all the policies. Given the first random choice of node, Greedy-AavgA can be "trapped" in the neighborhood of a node far from the best node to immunize, and thus can be forced to make a choice based on the limited local information.

Figure 6.10 demonstrates the effectiveness of our quality metric i.e. the λ_S value for each immunization policy after $k = 10$ immunizations. It is a scatter plot of Max. infections till steady state and the various λ_S values at the end of the immunizations. So points closer to the origin (minimum footprint and λ_S) represent better policies. Clearly, Optimal should theoretically be the closest to the origin (we don't show it as it didn't finish). Also as discussed before, Greedy-AavgA is the worst and that is demonstrated by its point. From Figure 6.9 we can see that Greedy-S has the least λ_S value after $k = 10$, hence it is closest to the origin and thus has the smallest footprint. Others perform well too, as their final λ_S values were close as well.

To summarize, in our experiments we demonstrated that policies decreasing λ_S the

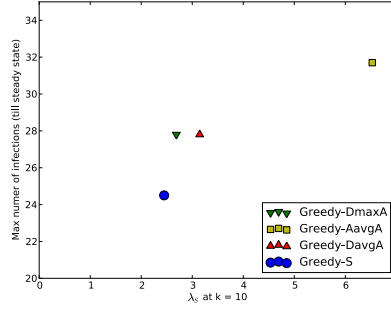


Figure 6.10: Scatter plot of Max. infections till steady state and λ_S for different immunization policies after $k = 10$ immunizations. Points closer to the origin are better. All policies perform in accordance to the λ_S values achieved (see Figure 6.9).

most are the best policies as they result in smaller footprints as well. Also, simple greedy policies were effective and achieved similar λ_S values like expensive combinatorial policies.

6.8.5 Discussion

We discuss some pertinent issues and give additional explanations in this section.

Goodness of the \mathbf{A}_{avg} matrix: We saw that Greedy-DavgA gave very good results and was close to Greedy-S and Optimal. This can be explained with the help of the following lemma.

Lemma 6.4. $(1 - 2\delta)\mathbf{I} + 2\beta\mathbf{A}_{\text{avg}}$ is a first-order approximation of the \mathbf{S} matrix.

Proof. Note that ($T = 2$),

$$\begin{aligned}
\mathbf{S} &= \mathbf{S}_1 \times \mathbf{S}_2 \\
&= ((1 - \delta)\mathbf{I} + \beta\mathbf{A}_1) \times ((1 - \delta)\mathbf{I} + \beta\mathbf{A}_2) \\
&= (1 - \delta)^2\mathbf{I} + (1 - \delta)\beta(\mathbf{A}_1 + \mathbf{A}_2) + \beta^2\mathbf{A}_1\mathbf{A}_2 \\
&\approx (1 - 2\delta)\mathbf{I} + \beta(\mathbf{A}_1 + \mathbf{A}_2) = (1 - 2\delta)\mathbf{I} + 2\beta \left(\frac{\mathbf{A}_1 + \mathbf{A}_2}{2} \right)
\end{aligned}$$

where we neglected second order terms involving β and δ . Thus $(1 - 2\delta)\mathbf{I} + 2\beta\mathbf{A}_{\text{avg}}$ is a first-order approximation of the \mathbf{S} matrix. \square

In other words, we can consider the time-varying system to be approximated by a static graph system of the \mathbf{A}_{avg} graph adjacency matrix with a virus of the same strength (β/δ remains the same). The threshold for a static graph with adjacency matrix \mathbf{A} is $\beta\lambda_{\mathbf{A}}/\delta$. So in our static case, we should aim to reduce $\lambda_{\mathbf{A}_{\text{avg}}}$ as much as possible. Any policy which aims to reduce the $\lambda_{\mathbf{A}_{\text{avg}}}$ value will approximate our original goal of dropping the λ_S value. Thus, this gives a theoretical justification of why Greedy-DavgA gave good results.

Temporal Immunization: In this section, we concentrated only on *static* immunization policies - policies where once immunized, a node is ‘removed’ from the contact graphs. While this makes sense for biological vaccinations, in a more complex ‘resource’ oriented scenario where each ‘glove’ protects a person only for the time it is worn, a time-varying immunization policy might be more useful. e.g., we may have finite number of gloves to give and we can change the assignment of gloves during day/night. In this case, it may be better to immunize nurses in hospitals during the day by giving them the gloves but during the night, we can decide to give gloves to restaurant waiters or children, because the nurses are now not well-connected in the contact graph. Our threshold results can trivially estimate the impact or any ‘what-if’ scenarios w.r.t. such temporal immunization algorithms.

6.9 Conclusion

We studied the problem of immunization (node-removal): both on graphs which are fixed and graphs which change with time in this chapter. Simply put we asked the question: "what are the best-k nodes to immunize to defend against an epidemic on static and time-varying graphs?". This problem is closely dependent on the epidemic threshold problem which we addressed in the previous chapters. Besides the problem definitions, our main contributions are:

1. A novel definition of ‘*Shield-value*’ score for a fixed graph, $Sv(S)$ for a set of nodes S , by carefully using the results from the theory of matrix perturbation; ($Sv()$ is a good approximation to the eigen-drop $\Delta\lambda$, the reduction of ‘*Vulnerability*’, see Lemma 6.1).
2. For the static case, we gave a *near-optimal* and *scalable* algorithm (NETSHIELD) to find a set of nodes with the highest ‘*Shield-value*’ score, by showing that our setting has the sub-modularity property (i.e., Theorem 6.1). Moreover, NETSHIELD also scales linearly with the size of the graph (number of edges). We also developed several heuristics for time-varying graphs.
3. Extensive experiments on several real data sets, illustrating both the effectiveness as well as the efficiency of our methods, sometimes outperforming competitors by several *orders of magnitude*.

A promising research direction is to parallelize the current methods (e.g., using Hadoop⁸).

⁸<http://hadoop.apache.org/>

Chapter 7

Fractional Immunization

In the previous chapter, we studied the problem of immunization as completely *removing* nodes from the network. While preventing contagion in networks is an important problem in public health and other domains, the assumption that selected nodes can be rendered completely immune does not hold for infections for which there is no vaccination or effective treatment. Instead, one can confer fractional immunity to some nodes by allocating variable amounts of infection-prevention resource to them. We formulate the problem to distribute a fixed amount of resource across nodes in a network such that the infection rate is minimized, prove that it is NP-complete and derive a highly effective and efficient linear-time algorithm. We demonstrate the efficiency and accuracy of our algorithm compared to several other methods using simulation on real-world network datasets including US-MEDICARE and state-level interhospital patient transfer data. We find that concentrating resources at a small subset of nodes using our algorithm is up to *6 times* more effective than distributing them uniformly (as is current practice) or using network-based heuristics. To the best of our knowledge, we are the *first* to formulate the problem, use truly nation-scale network data and propose effective algorithms.

7.1 Introduction

Given a graph and vaccines which provide partial ('fractional') protection, how to distribute them to maximize lives saved? Networks carry harmful agents, e.g., disease, computer viruses, and even misinformation. The networks' structure dictates how rapidly the malicious agent will spread. One can take advantage of this structure to identify specific nodes for infection control, such that the spread of the disease is significantly curtailed. In selecting nodes for infection control, previous work has assumed that nodes can be rendered completely immune. However, in many cases the complete immunization of a node is not an option. Bacteria present in hospitals have developed resistance to most antibiotics. Vaccines take time to be developed for both human and computer viruses, prompting other measures to prevent epidemics. However, one can provide

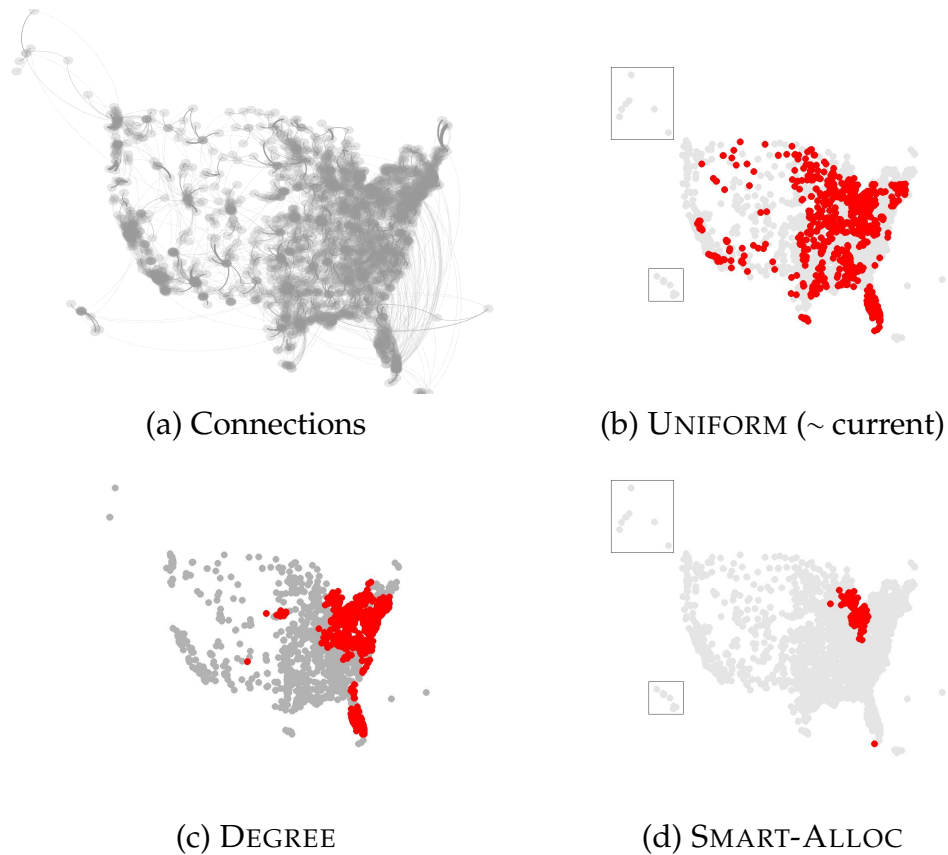


Figure 7.1: Our proposed SMART-ALLOC method minimizes number of infections (red circles): (a) The US-MEDICARE network overlaid on a map (b-d) Infected hospitals after a year (365 days) under different immunization algorithms. The same amount of resources ($k = 200$) were distributed by the algorithms. UNIFORM is the largely current practice of distributing evenly across all hospitals, while DEGREE distributes according to the number of connections of a hospital.

partial (“fractional”) protection by allocating resources to render nodes less susceptible.

In this chapter we formulate the problem of distributing resources to minimize the spread of infection on a network. Previously devised models, which assume that allocating a single unit of resource to a node renders it completely immune are a special case of this more general problem. We illustrate the problem in two settings: the spread of infection between hospitals through patient transfers, and the spread of malicious code between individuals through transfers of computer code between users in an electronic setting.

Consider the problem of prevention of hospital-to-hospital transfer of drug resistant bacteria. Critically ill patients are frequently and routinely transferred between hospitals in order to provide necessary specialized care [ICM⁺09]. While such interhospital transfers are an essential part of routine patient care, they also enable the transfer from

hospital to hospital of highly virulent micro-organisms resistant to many or all antibiotics [DWG10, Liv03]. As an example, recent work [Mea12] implicates inter-hospital patient transfers as an important vehicle for spread of “superbug” MRSA (methicillin-resistant *Staphylococcus aureus*). There is no existing technology, short of isolating a hospital, that can completely prevent the spread of such micro-organisms. To disrupt transfers by removing a hospital from the system can only be done under truly extraordinary circumstances (such as the outbreak of SARS in Toronto). Instead, there are large numbers of infection control technologies (e.g., bottles of disinfectant) that offer partial prevention and can be applied at individual hospitals (e.g., [ZIM⁺09]). Since such infection control technologies are costly, how should policy-makers optimally deploy them in order to minimize the global interhospital spread of highly resistant micro-organisms via patient transfers?

We also consider the spread of malicious content in an electronic setting. In the Second Life virtual world, nearly all content, from the landscape, to clothing, to the avatars’ movements, are created and distributed as scripts by the users themselves. This is part of the interactivity that has made the enterprise a success. However these virtual environments create the potential for malicious scripts to be inadvertently picked up and dispersed by unwitting users ¹. Without shutting down a users’ account, it is not possible to confer complete immunity upon that node. However, one could allocate resources differentially to a subset of nodes, in the form of educating users and auditing their code inventories.

Motivated by the above applications, and the many other instances where complete immunization is not feasible (e.g., HIV transmission, or H1N1 flu transmission prior to availability of vaccine) we study the problem of effective and efficient fractional immunization on directed weighted graphs. In fractional immunization, one allocates differing amounts of resource to nodes from a fixed total budget. Nodes which receive more infection-prevention resource have a smaller likelihood of becoming infected when exposed than nodes receiving no or little resource. A straightforward approach that tests each possible allocation would very quickly become computationally intractable (e.g., for a network with 2000 nodes, it will take more time than the age of the universe to examine all the possibilities on a 2GHz processor machine). Instead, we give an effective and efficient *linear-time* (in nodes and edges) algorithm SMART-ALLOC in this chapter. Our extensive experiments show that we may achieve significant benefits if nodes coordinate their allocation of resources, rather than making allocations independently, as is current practice in many settings. See Figure 7.1 for an illustration, where our algorithm outperforms other alternatives by up to 6x fewer infections.

The rest of the chapter is organized as follows: § 7.2 reviews related work, § 7.3 gives the problem formulation and the hardness result, § 7.4 and § 7.5 explain our proposed method and § 7.6 presents extensive experiments on datasets. Finally, we conclude in

¹<http://www.reuters.com/article/2007/08/20/us-disease-game-idusn2031054020070820?sp=true>

7.2 Related Work

In short, all the existing immunization strategies mentioned below assume: (1) *full* immunity - once a node is immunized, it is completely removed from the graph; (2) *binary* allocation (i.e., each node would need at most 1 antidote); and (3) *symmetric* immunization - once applied, an antidote affects both incoming and outgoing edges. These assumptions might be too strong for the inter-hospital patient transfers applications. To the best of our knowledge, we are the *first* to address the more realistic and challenging setting, where the effect of an antidote could be partial and asymmetric and the same node can receive multiple antidotes.

We review related work in the context of networks here, which can be categorized into three parts: epidemic thresholds, immunization algorithms and information diffusion.

Epidemic Thresholds/Conditions Much work has been done in finding epidemic thresholds (minimum virulence of a virus which results in an epidemic) for a variety of networks [Bai75, McK25, AM91, KW93, PSV02, WCWF03, GMT05, PCF⁺11, MNP06]. It should be pointed out that, with the exception of [MNP06] most if not all of the existing work assumes *symmetry* in virus propagation. That is, the probability that A infects B or B infects A is the same, assuming either A or B are infected. The inter-hospital patient transfer graph is *asymmetric*; a hospital that is better equipped to treat a critical care patient is more likely to be on the receiving end of a transfer. Asymmetries in transfer are also present in e.g. email networks.

Immunization Cohen et al. [CHbA03] studied the *acquaintance* immunization policy, and showed that it is much better than random, for both the SIS as well as the SIR model on random power-law graphs. Hayashi et al. [HMM03] modeled e-mail viruses using the SHIR model (Susceptible, Hidden, Infectious, Recovered) and derived the extinction conditions under random and targeted immunization. Tong et al. [TPT⁺10] proposed an effective immunization strategy in the SIS model also motivated by preventing the spread of computer viruses. Briesemeister et al. [BLP03] studied immunization policies on power-law graphs. Lappas et al. [LTGM10] study the problem of finding best-possible ‘culprits’ who started an infection.

General information diffusion processes There is a lot of research interest in studying dynamic processes on large graphs, (a) blogs and propagations [GGLNT04, KNRT03, KKT03, RD02], (b) information cascades [BHW92, GLM01, Gra78] and (c) marketing and product penetration [Rog03, LAH06]. These dynamic processes are all closely related to virus propagation. For example, one may wish to allocate third-party “fact checking” resources to content posted on specific blogs in order to minimize the spread

of misinformation in the network. Although no blog could be completely immune to spreading misinformation, such efforts would dampen its spread.

7.3 Problem Formulation and Hardness result

We first formulate the problem explicitly. Let A be the adjacency matrix of the connected directed weighted graph (of N nodes and E edges) on which the virus is spreading—entry $A(i, j)$ in the matrix denotes the weight on the edge between nodes (hospitals) i and j (e.g., the weight can be the average number of patient-transfers per day). The infection spreading model can be described by a flu-like model with no immunity, technically the SI model (‘susceptible-infected’) of epidemiology [AM91]. Briefly, at every time-step, any healthy node can get the infection from one of its currently infected neighbors. The probability of becoming infected by any particular neighbor during a period of time is independent and proportional to the weight of the connecting edge. Also, once infected, a node always stays infected. Any node gets partial immunity upon getting an antidote. Any amount x of antidote cuts the transmissibility of the virus by a factor $f(x)$ (called the utility function). For example, under function $f(x) = 0.5^x$, each additional antidote to hospital i decreases the probability of transmission from any neighbor j of i by a fixed percentage (50%). Our results hold for any utility function $f(x)$ with a diminishing marginal returns property typical of infection-control techniques (c.f. [ZIM⁺09]). Also note the inherent *asymmetric* nature of the effect of an antidote, it only effects the incoming edges of any node. The infection starts with some initially infected ‘seed’ nodes. We want to distribute the antidotes so that the expected “footprint” (the expected number of infections at some given time t) is minimized. To summarize, we are given:

- The SI model as the virus-spreading process
- A fixed directed weighted graph A (each edge e having weight w_e with $0 < w_e \leq 1$ e.g., the weight can be the average number of patient-transfers per day between hospitals)
- A total of k antidotes having partial effect e.g., bottles of disinfectant
- A weakening function $f(x)$, denoting how beneficial are x units of antidote, typically with diminishing marginal returns property

Using popular epidemiological assumptions, we assume that the virus and the underlying graph do not change.² The problem can be stated as:

Problem 7.1 (MAX-HEALTH). *Distribute the antidotes such that for an infection process spreading over the resulting graph after the antidote-allocation, we minimize the “footprint” (the expected number of infections at time t , for some given t).*

The current practice in allocating varying amounts of antidote across a network is effectively uniform, e.g. hospitals independently tackle infection control. However, this

²Relaxing these assumptions is a promising research direction.

makes no use of the connected network we are given. As mentioned in the introduction, another obvious but computationally prohibitively expensive method is to estimate the footprint through computer simulations. How can we get a practical and effective algorithm?

7.3.1 Our proposed problem—MIN-CONN

Main Idea Our observation is that the footprint depends on the connectivity of the underlying network and as we show next, the best single measure of connectivity is λ_A , the so-called ‘largest eigenvalue’ of the adjacency matrix of the network. Roughly, it describes the number of paths between pairs of nodes in a graph, discounting for longer paths. Earlier results [WCWF03, GMT05, PCF⁺11] have also shown that the epidemic threshold (maximum virus strength so that there is no epidemic) on unweighted, undirected graphs depends on the largest eigenvalue of the adjacency matrix. Instead of MAX-HEALTH, we then propose to minimize the largest eigenvalue of the weighted adjacency matrix while distributing the antidotes.

Justification Unfortunately, note that unlike other models, our virus spreading model is SI and thus has *no* epidemic threshold - any initial infection will eventually infect *everyone* in the graph. But still, as our next lemma shows, we can upper-bound the expected number of infected nodes in the graph at any time t :

Lemma 7.1. *In the SI virus spreading model on a graph:*

$$\sigma(t) \leq (1 + \lambda_A)^t \sigma(0)$$

where $\sigma(t)$ is the expected num. of infected nodes at time $t > 0$ and $\sigma(0)$ is a scalar depending just on the initial conditions (independent of t).

Proof. Suppose the discrete-time SI process is running on graph A and $p_i(t)$ denotes the probability that node i is infected at time t after the process started. Then,

$$p_i(t+1) = p_i(t) + (1 - p_i(t)) \cdot \Gamma_i \tag{7.1}$$

where Γ_i is the probability that node i receives some infection from any of its infected neighbors during the time t to $t+1$. Let R be an indicator random variable for the event that node i gets the infection during t to $t+1$. Clearly,

$$R = \mathbb{1}_{\cup_{j \in \text{neighbor}(i)} T_j}$$

where T_j is the event that node j transferred an infection between time t and $t+1$; $\mathbb{1}_j(t)$ is the corresponding indicator random variable. Using the well-known relation that expectation of an indicator random variable is just the p.d.f. of the random variable:

$$\begin{aligned} \Gamma_i = \mathbb{E}[R] &= \mathbb{E}[\mathbb{1}_{\cup_{j \in \text{neighbor}(i)} T_j}] \\ &\leq \sum_{j=1}^N \mathbb{E}[\mathbb{1}_j(t)] = \sum_{j=1}^N A(j, i) p_j(t) \end{aligned}$$

where the second step follows because for any two events A and B , $\mathbb{1}_{A \cup B} = \mathbb{1}_A + \mathbb{1}_B - \mathbb{1}_A \mathbb{1}_B \Rightarrow \mathbb{E}[\mathbb{1}_{A \cup B}] \leq \mathbb{E}[\mathbb{1}_A] + \mathbb{E}[\mathbb{1}_B]$. Thus using Equation 7.1 and above:

$$p_i(t+1) \leq p_i(t) + (1 - p_i(t)) \sum_{j=1}^N A(j, i) p_j(t)$$

Letting $\vec{P}(t) = [p_1(t), p_2(t), \dots, p_N(t)]^T$, we can write the entire system as:

$$\begin{aligned} \vec{P}(t+1) &\leq \vec{P}(t) + [I - \text{diag}(\vec{P}(t))] \times A^T \times \vec{P}(t) \\ &= \vec{P}(t) + A^T \vec{P}(t) - \text{diag}(\vec{P}(t)) A^T \vec{P}(t) \\ &\leq (I + A^T) \vec{P}(t) \\ &\leq (I + A^T)^t \vec{P}(0) \end{aligned}$$

Consider the all ones vector \vec{e} . Then for any $t > 0$, $\vec{e}^T \vec{P}(t) = \sigma(t)$, the expected number of infected nodes at time t . Hence,

$$\begin{aligned} \sigma(t+1) &\leq \vec{e}^T (I + A^T)^t \vec{P}(0) \\ &= \vec{e}^T \left(\sum_{j=1}^N (1 + \lambda_{A, j})^t \vec{v}_j \vec{u}_j^T \right) \vec{P}(0) \\ &\leq (1 + \lambda_{A, 1})^t \vec{e}^T \left(\sum_{j=1}^n \vec{v}_j \vec{u}_j^T \right) \vec{P}(0) \end{aligned}$$

where we used the spectral decomposition of matrix $I + A^T$ in the second step. Denoting $\lambda_{A, 1}$ as λ_A , we have that

$$\sigma(t+1) \leq (1 + \lambda_A)^t \sigma(0)$$

where $\sigma(0) = \vec{e}^T \left(\sum_{j=1}^n \vec{v}_j \vec{u}_j^T \right) \vec{P}(0)$ (a scalar depending just on the initial conditions independent of t). \square

Thus, we propose to minimize the upper-bound on the expected number of infected nodes at any time t , by minimizing the largest eigenvalue λ_A .

We call our proposed problem MIN-CONN. Suppose the vector which gives us the distribution for k antidotes is $\vec{m} = \{m_1, m_2, \dots, m_N\}$ (where m_i is the number of antidotes given to node i), with the constraint that $\sum m_i = k$. Denoting A' as the resulting adjacency matrix after distributing the antidotes, our transformed problem can be stated as:

Problem 7.2 (MIN-CONN). *Distribute the antidotes such that the largest eigenvalue of the resulting adjacency matrix is minimized i.e.*

$$\text{minimize } \lambda_{A'} \quad \text{s.t.} \quad \sum_i m_i = k, \quad \forall_i m_i \in \{0, 1, \dots\}$$

It is easy to see that if we define a matrix $F = \text{diag}(f(\vec{m}))$,³ then $A' = A \times F$.

³ $f(\vec{m})$ just applies the function f on each element of the vector \vec{m}

7.3.2 MIN-CONN is NP-complete

Unfortunately, MIN-CONN is NP-complete. Consider the decision version of MIN-CONN:

Problem 7.3 (MIN-CONN Decision Version). *Given a directed and weighted graph $G = (V, E)$, $k \geq 0$, $t \geq 0$, and non-increasing $f(x)$ (hence, instance $(G, k, t, f(x))$) is there an assignment \vec{m} with $\sum_i m_i = k$, $\forall_i m_i \in \mathbb{Z}^*$ such that $\lambda_{AF} \leq t$ where A is the adjacency matrix of G and $F = \text{diag}(f(\vec{m}))$?*

We will prove MIN-CONN (Decision version) is NP-complete next.

Theorem 7.1. *MIN-CONN (Decision Version) is NP-complete.*

Proof. Clearly, MIN-CONN (Decision Version) is in NP: given an integral assignment \vec{m} as witness, we can check in poly-time if the largest eigenvalue is less than the threshold. Hence we just need to prove that it is poly-time reducible from an NP-complete problem.

We reduce from INDEPENDENT-SET, a well-known NP-complete problem [GJ83].

Problem (INDEPENDENT-SET). *Given a undirected, unweighted graph $G = (V, E)$ and a number $k > 0$ (i.e. instance (G, k)), is there a set of k vertices, no two of which are adjacent?*

Say the size of G is n . Given an instance of INDEPENDENT-SET (G, k) we create an instance $I \equiv (G, n - k, 0, f(x))$ of MIN-CONN where $f(x)$ is defined as

$$f(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{if } x > 0 \end{cases}$$

Note that such a $f(x)$ forces any algorithm for MIN-CONN to essentially choose k vertices whose all incoming edges will be deleted. Clearly this construction takes polynomial time. We now need to prove two things:

1. If there is an independent set S in G , the instance I has a YES answer.

This is true, because we can set $m_i = 1$ for all $n - k$ nodes i *not* in S (i.e. $V \setminus S$). Consider the resulting graph G' . There will not be any edges from vertices in S to any other vertex. Also, there will not be any edges from vertices in set $V \setminus S$ to each other. These follow because of the antidote distribution and the fact that S was an independent set for G . Hence, the adjacency matrix AF of G' will look like:

$$AF = \begin{bmatrix} 0_{n-k, n-k} & C \\ 0_{k, n-k} & 0_{k, k} \end{bmatrix}$$

where C is a size $(n - k) \times k$ matrix representing the edges from $V \setminus S$ to S . It is easy to check that the largest eigenvalue of AF is 0. Hence I has a YES answer.

2. If G does not have an independent set of size k , then instance I has a NO answer.

Suppose the algorithm for MIN-CONN selects $n - k$ vertices whose all incoming edges will be deleted. Call the un-selected vertices set S ($|S| = k$) and the resulting graph

G' (adjacency matrix AF). Consider G_S and G'_S , the subgraph induced by the vertices of S in G and G' respectively. Clearly $G_S \equiv G'_S$, as the algorithm didn't select any vertex in S . Also, as G does not have an independent set of size k , G_S is not a null graph (with no edges) and thus has some connected sub-graph H . Applying the Perron-Frobenius theorem [McC00], the largest eigenvalue of the adjacency matrix for H is positive. Denoting the adjacency matrix of G'_S (or G_S) as D , the matrix AF will look like:

$$AF = \begin{bmatrix} 0_{n-k, n-k} & C \\ 0_{k, n-k} & D \end{bmatrix}$$

where like before C is a size $(n - k) \times k$ matrix representing the edges from $V \setminus S$ to S . We know that the largest eigenvalue of AF is at least the largest eigenvalue of D and the largest eigenvalue of D is at least the largest eigenvalue of the adjacency matrix of H (eigenvalue interlacing). Hence, D has at least one non-zero eigenvalue. Thus for any S , the largest eigenvalue of AF is non-zero and hence instance I has a NO answer.

Hence, MIN-CONN (Decision version) is NP-complete. \square

7.4 Proposed Method—Overview

As MIN-CONN is NP-complete, we resort to heuristics. A simple and intuitive heuristic is to disrupt the connectivity of the network by distributing the antidotes according to the number of neighbors ('degree') of a hospital. Thus a hospital involved in a larger number of total patient transfers will receive more resources than small isolated hospitals. This appears to be a reasonable approach until we realize that this does not directly attack the exact connectivity metric: λ_A . For example, this method will allocate most of the resources to the big coastal hospitals, and may miss out on a critical but mid-sized central hospital acting as a 'bridge' between the coasts. Hence, our heuristic should directly try to optimize the drop in λ_A . Next we present two such heuristics in improving speed: (a) EXHAUSTIVE, (b) SMART-ALLOC.

7.4.1 Algorithm EXHAUSTIVE

Algorithm EXHAUSTIVE greedily tries to find the best hospital to allocate each additional antidote to. Clearly, the best node is the one in the graph which, when given the extra antidote, decreases λ_A the most at that step. Hence, we need to compute the largest eigenvalue N times for making only a *single* allocation decision (so for k antidotes, it will be done $k \times N$ times). This is very expensive e.g., for a US-wide network of about 2000 hospitals, it took about a *day* to distribute only 1500 antidotes. The total running time would be $O(kNE)$, using the $O(\#edges)$ Lanczos algorithm for computing the eigenvalue). For larger graphs (such as our Second-Life network), this would be too slow to be feasible.

7.4.2 Algorithm SMART-ALLOC

We give an overview of our approach here, and the theoretical under-pinnings in the next section.

Best single allocation Following from the discussion above, instead, we can give each additional antidote to the currently most ‘important’ (central) hospital, with the hope that it is also the hospital reducing λ_A the most. Fortunately, we can show that the measure of centrality which allows us to closely approximate the drop in λ_A is the so-called Eigenvector centrality adapted to directed graphs (a combination of the so-called ‘hub’-ness and ‘authority’-ness scores [Kle98] of each node). We just give the next antidote to the hospital which has the highest such centrality score currently. This would be faster than EXHAUSTIVE, though with some approximation. Note that we still have to perform the eigenvalue computation (to update the centralities of all the nodes) after each allocation decision. Can we do better?

Batched allocation The answer is yes - in fact, we can make r times fewer updates (for a suitably chosen r) to node centralities by carefully allocating r antidotes in one go, using *only* the old centrality values. Thus we need to update and ‘resync’ the centralities only every r allocations. We call this algorithm SMART-ALLOC: it is much faster (linear on number of nodes and edges) than the other methods with minimal loss of accuracy at the same time (a point we illustrate using experiments as well—see Sections 7.6.2 and 7.6.3).

7.5 Proposed Method—Theorems and proofs

Here we give details of the two main ideas we mentioned above. Jumping ahead, our effective and efficient algorithm SMART-ALLOC is given in § 7.5.2.

7.5.1 Best single allocation—Details

Let $\vec{u} = [u_1, u_2, \dots, u_N]^T$ and $\vec{v} = [v_1, v_2, \dots, v_N]^T$ be right and left eigenvectors of A corresponding to λ_A . In a nutshell, the best node i^* to give a single antidote is the one with the maximum value of $u_i v_i$ i.e. $i^* = \arg \max_i u_i v_i$. We can prove the following lemma to justify it.

Lemma 7.2. *Assuming the current adjacency matrix is A , the change in the in the largest eigenvalue $\Delta\lambda_A$ after distributing one antidote to a node, say i , approximated to the first order is given by:*

$$\Delta\lambda_A = \lambda_A \left(\frac{f(1)u_i v_i}{v^T u} - 1 \right)$$

Proof. We know that $A\mathbf{u} = \lambda_A\mathbf{u}$ and $\mathbf{v}^\top A = \lambda_A\mathbf{v}^\top$ (right and left eigenvectors). According to the Perron-Frobenius theorem [McC00], λ_A is real and non-negative and the components of the corresponding eigenvectors \mathbf{v} and \mathbf{u} all are positive. After a small modification due to the medicine:

$$(A + \Delta A)(\mathbf{u} + \Delta\mathbf{u}) = (\lambda_A + \Delta\lambda_A)(\mathbf{u} + \Delta\mathbf{u})$$

Premultiplying by \mathbf{v}^\top and neglecting second order terms:

$$\Delta\lambda_A \approx \frac{\mathbf{v}^\top \Delta A \mathbf{u}}{\mathbf{v}^\top \mathbf{u}} \quad (7.2)$$

Clearly, after distributing one antidote to node i , ΔA is:

$$\Delta A = AF_i - A \quad (7.3)$$

where $F_i = \text{diag}([f(0), \dots, f(1), \dots, f(0)])$ (i.e. the i -th position on the diagonal is $f(1)$). Using it in Equation 7.2:

$$\begin{aligned} \Delta\lambda_A &\approx \frac{\mathbf{v}^\top AF_i \mathbf{u}}{\mathbf{v}^\top \mathbf{u}} - \frac{\mathbf{v}^\top A \mathbf{u}}{\mathbf{v}^\top \mathbf{u}} \\ &= \frac{\lambda_A \mathbf{v}^\top F_i \mathbf{u}}{\mathbf{v}^\top \mathbf{u}} - \lambda_A \\ &= \lambda_A \left(\frac{f(1)u_i v_i}{\mathbf{v}^\top \mathbf{u}} - 1 \right) \end{aligned} \quad (7.4)$$

Proved. □

This requires the computation of \vec{u} and \vec{v} , which is $O(E)$. We can continue giving the antidotes in this way, but as discussed above, we will need to re-compute \vec{u} and \vec{v} after each allocation decision.

7.5.2 Batched allocation—Details

In sum, SMART-ALLOC uses Algorithm 2 to batch-allocate and resync till we have exhausted total budget k (see § 7.5.2). We now show how we can batch-allocate r antidotes in one-go. Suppose the distribution of allocations as before is given by the vector \vec{m} . In this case, we can prove the following lemma, similar to Lemma 7.2.

Lemma 7.3. *The change in the largest eigenvalue $\Delta\lambda_A$ after distributing r antidotes according to \vec{m} (so $\sum_i m_i = r$) approximated to the first order is given by:*

$$\Delta\lambda_A = \lambda_A \left(\frac{\mathbf{v}^\top F \mathbf{u}}{\mathbf{v}^\top \mathbf{u}} - 1 \right)$$

where \mathbf{v} and \mathbf{u} are the left and right eigenvectors of A corresponding to λ_A and $F = \text{diag}(f(\vec{m}))$.

Proof. (Details Omitted for brevity) The main change from Lemma 7.2 is that $\Delta A = AF - A$ now. \square

Subsequently, for the best allocation of r antidotes, it is easy to see that we have the following optimization problem now, analogous to MIN-CONN:

Problem 7.4 (MAX-DROP). *Distribute antidotes such that:*

$$\text{minimize } \sum_{i=1}^N f(m_i) \cdot u_i \cdot v_i \quad \text{s.t.} \quad \sum_i m_i = r$$

(of course, $\forall_i m_i \in \{0, 1, \dots\}$). Clearly, it is an integer optimization problem, which in general is NP-complete.

GreedyDrop: An optimal greedy algorithm

Surprisingly, we can prove that a greedy algorithm achieves the optimal solution for MAX-DROP, when $f(x)$ is *monotone non-increasing convex*. The algorithm is given in Algorithm 2.

Algorithm 2 *GreedyDrop*

Input: Directed Weighted Adjacency matrix A , batch-size r , monotone non-increasing convex function $f(x)$

- 1: u = first right eigenvector of A
 - 2: v = first left eigenvector of A
 - 3: $\vec{m} = \vec{0}$
 - 4: **for** $i = 1$ to r **do**
 - 5: $j = \max_h [f(m_h) - f(m_h + 1)]u_h v_h$
 - 6: $m_j = m_j + 1$
 - 7: **end for**
 - 8: **return** \vec{m}
-

Intuitively, at each iteration, we pick the index (node) j which maximizes the drop in the value of the objective *at that* step. Clearly, the running time of the algorithm is $O(E + kN)$. We prove the optimality of *GreedyDrop* in Theorem 7.2. First, we prove the following lemma:

Lemma 7.4. *Given a convex non-increasing function $f(x)$, define function $g(x) = f(x) - f(x+1)$. Then $g(x)$ is non-increasing.*

Proof. As $f(x)$ is monotone non-increasing and convex, from the property of convex functions:

$$f(x) - f(y) \geq f'(y)[x - y] \quad \forall x, y \tag{7.5}$$

Using Equation 7.5 with $x = x, y = x + 1$ and $x = x + 1, y = x$, we get:

$$-f'(x + 1) \leq g(x) \leq -f'(x)$$

Similarly,

$$-f'(x + 2) \leq g(x + 1) \leq -f'(x + 1)$$

Clearly, from the preceding inequalities, we have that $\forall x \ g(x + 1) \leq g(x)$ i.e. $g(x)$ is a non-increasing function. \square

Theorem 7.2. *GreedyDrop returns the optimal integral \vec{m} for MAX-DROP when $f(x)$ is monotone non-increasing and convex.*

Proof. Say *GreedyDrop* returns m^G as the answer, but m^* is the true optimal. Then there was some first step (say t) where we incremented some m_j from s_j to $s_j + 1$ in m^G but m^* has $m_j = s_j$. Because we have a fixed batch-budget r , m^* also has some m_k as $s_k + 1$ while m^G has m_k which is *at most* s_k .

Consider another assignment m' which is identical to m^* except $m_k = s_k$ and $m_j = s_j + 1$. Note that we are still satisfying our constraint and hence it is a valid assignment. The score of this assignment is:

$$\begin{aligned} \text{Score}(m') &= \sum_{i=1}^N f(m_i) \cdot u_i \cdot v_i \\ &= \text{Score}(m^*) + [f(s_k) - f(s_k + 1)]u_k v_k \\ &\quad - [f(s_j) - f(s_j + 1)]u_j v_j \end{aligned} \tag{7.6}$$

where the last step is due to the construction of m' .

Recall that while computing m^G , *GreedyDrop* had chosen j at step t i.e.,

$$j = \max_h [f(m_h) - f(m_h + 1)]u_h v_h$$

at step t . At that instant, suppose $m_k = s'_k$. Hence from the above equation we can conclude that:

$$[f(s'_k) - f(s'_k + 1)]u_k v_k \leq [f(s_j) - f(s_j + 1)]u_j v_j \tag{7.7}$$

In addition, we know that $s'_k \leq s_k$. But from Lemma 7.4, $g(s'_k) \geq g(s_k)$ i.e.

$$f(s'_k) - f(s'_k + 1) \geq f(s_k) - f(s_k + 1) \tag{7.8}$$

So, from Equations 7.7 and 7.8:

$$[f(s_k) - f(s_k + 1)]u_k v_k \leq [f(s_j) - f(s_j + 1)]u_j v_j$$

Coupled with Equation 7.6, the above inequality implies that $\text{Score}(m') \leq \text{Score}(m^*)$. If $\text{Score}(m') < \text{Score}(m^*)$, then m^* is not optimal as we started with the assumption that m^* is optimal and hence has the lowest score. If $\text{Score}(m') = \text{Score}(m^*)$, then we can conclude that *GreedyDrop* did not make an error at step t and made it at some other point. Continuing similarly, finally, either m^* is not optimal or *GreedyDrop* is correct. Hence, a contradiction, m^G is optimal and *GreedyDrop* gives the optimal integral answer. \square

SMART-ALLOC

Finally, we are ready to describe our algorithm SMART-ALLOC: use *GreedyDrop* (Algorithm 2) to batch-allocate a small number (r) of resources and then ‘re-sync’ (re-compute) the first left and right eigenvectors and continue similarly till our budget k is exhausted.

One may ask why can not we directly allocate all k antidotes in one-go using *GreedyDrop*? This is because, unfortunately, the accuracy of the first-order approximation in Lemma 7.3 is only good when the number of antidotes k is small w.r.t. the graph i.e when $k \ll N$. But that is not the case in general - for e.g. in our motivating problem one may want to distribute 200 infection control resources among 2000 nationwide hospitals. In fact, k can be arbitrarily high, since the units of resource in this problem are set with arbitrary granularity. It is easy to see the next lemma:

Lemma 7.5 (Running time of SMART-ALLOC). *The running time of the algorithm SMART-ALLOC is $O(kE/r + kN)$.*

Clearly, we want to use as large r as possible. Our proposed rule-of-thumb is to choose r proportional to the spectral-gap ($|\lambda_A| - |\lambda_{A,2}|$) of the graph. Larger the spectral-gap, lesser is the sensitivity of the spectrum of A [GVL89], lesser is the need to re-sync often and hence larger is the r we can use e.g. in our experiments on hospital graphs, which had a small spectral-gap, we found that $r = 6$ performed very well.

7.6 Experiments

We designed experiments to answer the following questions about our algorithm SMART-ALLOC: (i) Effectiveness for reducing the rate of infection, (ii) Effectiveness for MIN-CONN and (iii) Scalability. In short, SMART-ALLOC proves to be a fast and effective algorithm for both reducing the rate of infection and solving MIN-CONN and is very close to EXHAUSTIVE, at a fraction of the running cost, while others are much worse.

7.6.1 Setup

For answering the above questions we ran extensive simulation experiments and compared against many other resource allocation methods (see Table 7.1) on multiple real-world datasets (see Table 7.2). We ran parallel experiments on a Condor [TTL05] cluster of 58 cores each being a generic Fedora 7 machine. All the algorithms and the SI infection process were coded in C++. We use $f(x) = 0.50^x$ and $r = 6$ for all our experiments. The choice of the function $f(x)$ captures the diminishing marginal utility of infection control based on a wide-range of studies in the medical literature of existing infection control techniques (c.f. [ZIM⁺09]).

Table 7.1: Various Algorithms used for comparison

Method Name	Method Description	Speed $O(\cdot)$
UNIFORM	Distribute uniformly among the nodes, breaking ties randomly.	kN
DEGREE	Distribute randomly proportional to the ‘degree’ [†] of the nodes.	$E + kN$
EXHAUSTIVE	Allocate each additional antidote to that node which decreases the largest eigenvalue λ_A the most in that step.	kEN
SMART-ALLOC	Allocate r antidotes in one go based on node centralities and only then recompute.	$kE/r + kN$

[†]As the graphs are directed, we use degree centrality [Fre79] - geometric mean of in-degree (the number of transfers the hospital receives) and out-degree (the number of transfers the hospital sends out).

Table 7.2: Various real-world datasets used in our work

Dataset Name	Nodes (N)	Edges (E)	Description
US-MEDICARE	2138	10241 [‡]	All critical patient transfers among US hospitals based on all Medicare Provider Analysis and Review (MedPAR) final action claims between Sept. 1, 2004 - Sept. 1, 2005 [ICM ⁺ 09].
PENN-ALL	137	1121 [‡]	Critical patient transfers within Pennsylvania hospitals based on all discharges (not just Medicare) between July 1, 2004 - June 30, 2006 [ICM ⁺ 09].
GESTURE	166,774	1.5 million	Second-Life transfer-network of ‘gestures’ among users. Gestures can include anything from animation, chat to playing sounds.

[‡] Weight for each edge $u \rightarrow v$ was the average number of transfers from hospital u to v per day.

7.6.2 Effectiveness for MIN-CONN problem

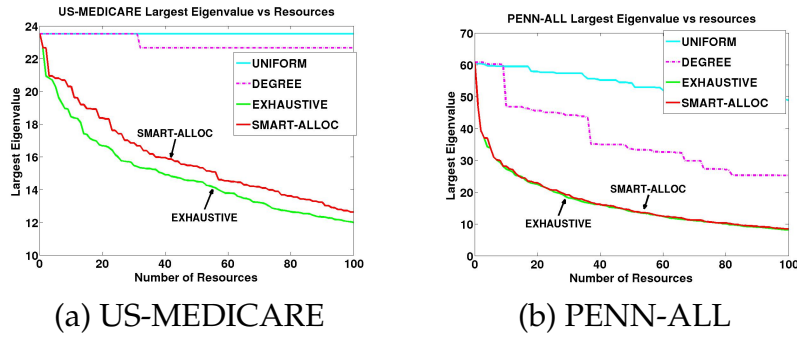


Figure 7.2: Largest Eigenvalue after allocation vs Budget k of resources used for different algorithms. (a) US-MEDICARE Network (b) PENN-ALL Network. Lower is better and SMART-ALLOC is near-optimal in both cases. (plots use color)

MIN-CONN aims to decrease the largest eigenvalue the most - how do the algorithms perform in that measure? In short, SMART-ALLOC comes very close to EXHAUSTIVE while others are much worse. Figure 7.2 shows the largest eigenvalue of the resulting graph after giving k antidotes according to various algorithms vs k on the US-MEDICARE and PENN-ALL networks. UNIFORM and DEGREE perform poorly, although DEGREE is better (sometimes marginally) than UNIFORM. SMART-ALLOC and EXHAUSTIVE are much better at achieving the lowest eigenvalue for all k . EXHAUSTIVE is expected to be near-optimal as it does an exhaustive search via repeated eigenvalue computation for the node which decreases the eigenvalue the most. On the other hand, SMART-ALLOC performs well due to our careful approximation and algorithm-design.

7.6.3 Effectiveness for MAX-HEALTH problem

We ultimately want to test how the algorithms perform for MAX-HEALTH. In short, again, SMART-ALLOC proves to be an effective algorithm and is very close to EXHAUSTIVE while others are much worse. See Figures 7.3 and 7.4 - they show the expected number of infected nodes (hospitals) vs. time tick after running the infection process on the partially immunized US-MEDICARE and PENN-ALL networks for different budget k of antidotes. The different curves represent the different algorithms used for allocation. As the edge-weights represent the average number of transfers per day, the curves represent the average footprint for each day after the infection starts. Each curve is an average of 21380 and 1370 simulation runs for US-MEDICARE and PENN-ALL respectively - in this way we ensured that we seeded the infection from each hospital for 10 different runs. We ran the simulations till 365 time-ticks (= 1 year) and took the average over all runs for each time-tick. Finally, the range of values of k for US-MEDICARE and PENN-ALL were chosen according to the network sizes and the function $f(x) = 0.50^x$.

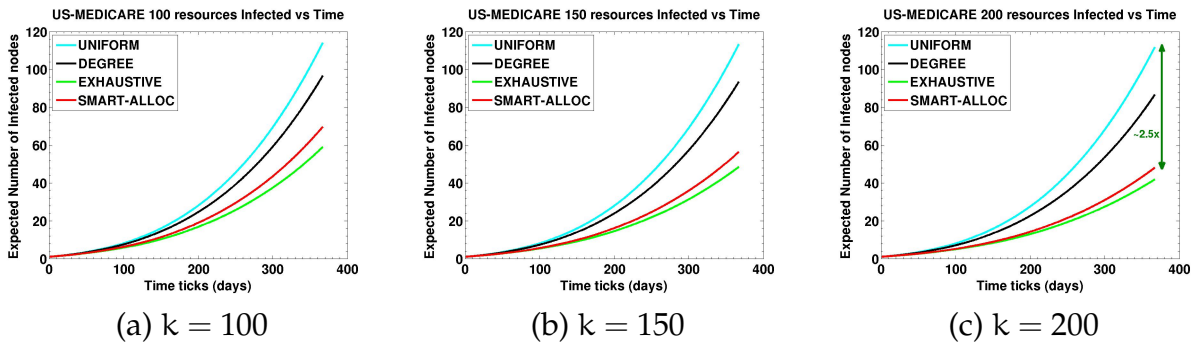


Figure 7.3: US-MEDICARE network for different algorithms and budget k of resources: Expected Number of Infections vs Time ticks (\approx days). Again EXHAUSTIVE and SMART-ALLOC perform the best and are close to each other, as expected. Each curve average of 21380 runs and lower is better (plot uses color)

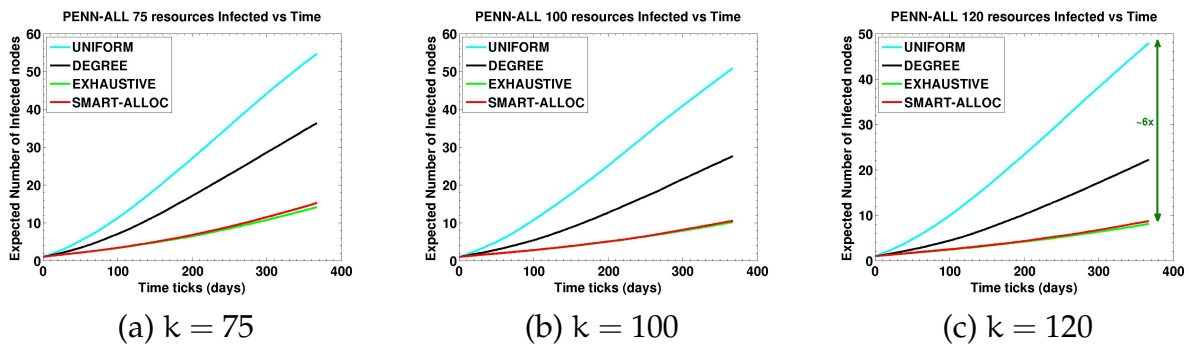


Figure 7.4: PENN-ALL network for different algorithms and budget k of resources: Expected Number of Infections vs Time ticks (\approx days). Again EXHAUSTIVE and SMART-ALLOC perform the best (they are almost on top of each other), as expected. Each curve average of 1370 runs and lower is better (plot uses color)

Our algorithm SMART-ALLOC clearly is very close to EXHAUSTIVE and has the lowest footprints everyday compared to the rest. For e.g., in Figure 7.3(f), after an year with $k = 200$ antidotes, EXHAUSTIVE and SMART-ALLOC have an expected total of 42 and 46 hospitals infected, while the other methods end with about 2.5 times worse at around 110. It is even more pronounced in PENN-ALL (Figure 7.4(f)): after an year with $k = 120$, EXHAUSTIVE and SMART-ALLOC have an expected footprint of ~ 8 , while the next closest method is about 3 times worse at around 23. This shows the dramatic impact an effective allocation algorithm can have on the number of infected nodes. Moreover note that all algorithms essentially mimic their performance w.r.t. MIN-CONN (Figure 7.2) i.e. larger the corresponding drop in the first eigenvalue λ_A , lower is the number of expected infections, validating our reduction of MAX-HEALTH to MIN-CONN.

The current practice is for each hospital to independently manage infection control,

which may be no better from the network perspective than using UNIFORM. But note that compared to UNIFORM, SMART-ALLOC can be up to 6 times better (see Figure 7.4(f)). Interestingly, for the US-MEDICARE network, we found that to achieve the same level of infection control as SMART-ALLOC and $k = 120$, we need a budget of about $k = 800$ resources if distributed according to UNIFORM.

7.6.4 Scalability

As discussed before, SMART-ALLOC is much faster than its chief competitor EXHAUSTIVE (see Table 7.1). For example, it took more than *10 hours* to distribute 200 resources using EXHAUSTIVE on the US-MEDICARE network while it took just *~ 14 seconds* to run SMART-ALLOC for the same budget - a 2500x speed-up. As a further comparison, the naïve simulation-based algorithm ran for a week and still had not finished for the same budget - a more than 30,000x speed-up. Additionally, on the GESTURE network, we had to stop EXHAUSTIVE after it took *3 days* to allocate a *single* resource; SMART-ALLOC took *~ 150 mins* to allocate *2000* resources.

7.6.5 Generality

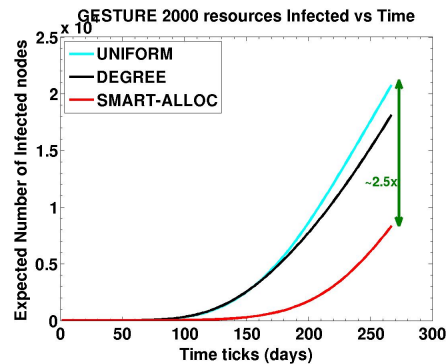


Figure 7.5: Expected number of infections vs time-ticks for different algorithms, budget $k = 2000$ on the GESTURE network. SMART-ALLOC is the best. Each curve average of 1000 runs. (plot uses color)

As mentioned in § 7.1, although our problem was originally motivated on hospital-transfer networks, the problem of fractional immunization arises in many other scenarios, and is arguably more realistic than complete immunization. To demonstrate the utility of SMART-ALLOC in domains other than epidemiology, we also compare performance on the GESTURE network of asset transfers between virtual world users. Figure 7.5 shows the the expected number of infected users vs. time, if a malicious asset were to be created and transferred between users. We budget $k = 2000$ antidotes, and select the infection source randomly. We don't show EXHAUSTIVE because, as mentioned before, it

didn't complete even after 3 *days* whereas SMART-ALLOC allocated all 2000 resources in ~ 150 mins. As expected, SMART-ALLOC has the fewest users infected, while others have up to ~ 2.5 times more users infected, demonstrating the efficacy of our algorithm in a completely different domain.

7.7 Conclusion

This work is the *first* to address the problem of allocation of infection-control resources with fractional and asymmetric impact among nodes in a network. It is a more general problem than that of selecting a subset of nodes to be immunized completely via a vaccine. The potential applications are broad—from curbing spread of infection between hospitals from patient transfers, to preventing spread of malicious code in virtual world settings.

We formulated the problem, proved it is NP-complete, and gave a highly efficient and effective algorithm SMART-ALLOC, which also we demonstrated through extensive experiments on multiple real-world datasets, including nation-wide patients-transfer networks and electronic virtual-world social transfer-networks. SMART-ALLOC runs in seconds (as opposed to weeks), on commodity hardware; more importantly, applied on real hospital-transfer networks (2005 U.S. Medicare data, 2004-2006 PA all-payer data) it results to up to $6x$ fewer infections, compared to current practice and other heuristics.

The current practice in control of highly resistant organisms via patient transfers has been largely focused within individual hospitals. Hence, the current public health policy is missing an opportunity to significantly reduce infection rates with an infection prevention strategy that accounts for the potential transfer of bacteria along the network of inter-hospital patient transfers.

Chapter 8

General Edge Placement

In this chapter, we shift the problem of controlling the dissemination of an entity (e.g., meme, virus, etc) on a large graph to the level of edges and ask: which edges should we add or delete in order to speed-up or contain a dissemination? First, we propose effective and scalable algorithms to solve these dissemination problems. Second, we conduct a theoretical study of the two problems and our methods, including the *hardness* of the problem, the *accuracy* and *complexity* of our methods, and the *equivalence* between the different strategies and problems. Lastly, we conduct experiments on real topologies of varying sizes to demonstrate the effectiveness and scalability of our approaches.

8.1 Introduction

As we have already seen in previous chapters, managing the dissemination of an entity (e.g., meme, virus, etc) on a large graph is a challenging problem with applications in various settings and disciplines. In its generality, the propagating entity can be many different things, such as a meme, a virus, an idea, a new product, etc. The propagation is affected by the topology and the properties of the entity: its ‘virality’, its speed, its ‘stickiness’ or the duration of the infection of a node. Our focus here is the *topology*, since we assume that we cannot alter the properties of the propagating entity.

The problem we address is how we can affect the propagation by modifying the *edges* of the graph. In fact, we address two different problems. First, in the **NETMELT problem**, we want to contain the dissemination by removing a given number of edges. For example, we can consider the distribution of malware over a social network. Deleting user accounts may not be desirable, but deleting edges (‘unfriending’ people) may be more acceptable. More specifically, we want to delete a set of k edges from the graph to minimize the infected population. Second, in the **NETGEL problem**, we want to enable the dissemination by adding a given number of edges. Specifically, we want to add a set of k new edges into the graph to maximize the population that adopt the information. For example, we could extend the social network scenario using the recent ‘arab spring’ which often used Facebook and Twitter for coordinating events: we may

want to maximize the spread of a potential piece of information. Note that an additional, key requirement for both problems is computational efficiency: the solution should scale to large graphs.

Both problems are challenging for slightly different reasons. For the NETMELT problem, most of the existing methods operate on the *node-level*, e.g., deleting a subset of the nodes from the graph to minimize the infected population from a propagating virus. In the above social spam example, this means that we might have to shutdown some legitimate user accounts. Can we avoid this by operating on a finer granularity, that is, only deleting a few edges between users to slow down the social spam spreading? For the NETGEL problem, things are even more challenging because of its high intrinsic time complexity. Let n be the number of the nodes in the graph. There are almost n^2 non-existing edges since many real graphs are very sparse. In other words, even if we only want to add one single new edge into the graph, the solution space is $O(n^2)$. This complexity ‘explodes’ if we aim to add multiple new edges collectively, where the solution space becomes *exponential*. To date, there does not exist *any* scalable solution for the NETGEL problem.

The overarching contribution of this work is the formulation and theoretical study of the dissemination management via edge manipulation: how to place a set of edges¹ to achieve the desired outcome. The main contributions can be summarized as follows:

- *Algorithms*. We propose effective and scalable algorithms to optimize the leading eigenvalue, the key graph parameter that controls the information dissemination processes for both NETMELT and NETGEL, respectively;
- *Proofs and Analysis*. We show the *accuracy* and the *complexity* of our methods; the *hardness* of the problem, and *equivalence* between the different strategies;
- *Experimental Evaluations*. Our evaluations on real large graphs show that our methods are both effective and scalable (see Fig. 8.1 as an example).

The rest of the work is organized as follows. We introduce notation and formally define the NETGEL and NETMELT problems in Section 2. We present and analyze the proposed algorithms in Section 3 and Section 4, respectively. We provide experimental evaluations in Section 5. We review the related work in Section 6 and conclude in Section 7.

8.2 Problem Definitions

Table 8.1 lists the main symbols used throughout the chapter. We consider directed, irreducible unipartite graphs. For ease of presentation, we discuss the unweighted graph scenario although the algorithms we propose can be naturally generalized to the weighted case. We represent a graph by its adjacency matrix. Following the standard notation, we use bold upper-case for matrices (e.g., \mathbf{A}), bold lower-case for vectors (e.g.,

¹In this work, we use the terms ‘link’ and ‘edge’ interchangeably.

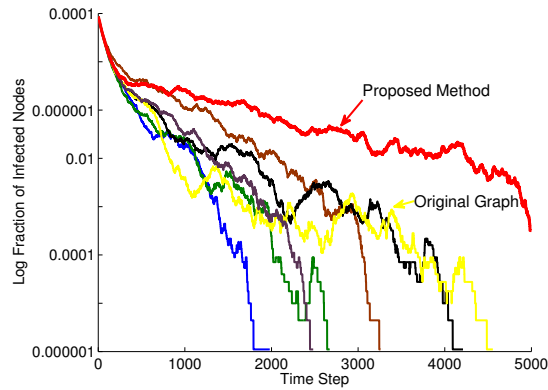


Figure 8.1: Comparison of maximizing the outcome of the information dissemination process. Larger is better. The proposed method (red) leads to the largest number of ‘infected’ nodes (e.g., having more people in the social networks to adopt a piece of good idea, etc). Notice that all the alternative methods are mixed with the result on the original graph (yellow), which means that they fail to affect the outcome of the dissemination process. See Section 6 for detailed experimental setting.

a), and calligraphic fonts for sets (e.g., \mathcal{J}). We denote the transpose with a prime (i.e., \mathbf{A}' is the transpose of \mathbf{A}). Also, we represent the elements in a matrix using a convention similar to Matlab, e.g., $\mathbf{A}(i, j)$ is the element at the i^{th} row and j^{th} column of the matrix \mathbf{A} , and $\mathbf{A}(:, j)$ is the j^{th} column of \mathbf{A} , etc.

When we discuss the relationship between the two different strategies (node deletion vs. edge deletion) for the NETMELT problem, it is helpful to introduce the concept of line graph, where the nodes represent the edges in the original graph. Formally, each edge in the original graph \mathbf{A} becomes a node in the line graph $L(\mathbf{A})$; and there is an edge from one node to the other in the line graph if the target of the former edge is the same as the source of the latter edge in the original graph \mathbf{A} . It is formally defined as follows:

Definition 8.1 (Line Graph). *Given a directed graph \mathbf{A} , its directed line graph $L(\mathbf{A})$ is a graph such that each node of $L(\mathbf{A})$ represents an edge of \mathbf{A} , and there is an edge from a node e_1 to e_2 in $L(\mathbf{A})$ iff for the corresponding edges $\langle i_1, j_1 \rangle$ and $\langle i_2, j_2 \rangle$ in \mathbf{A} , $j_1 = i_2$.*

With the notation of the line graph $L(\mathbf{A})$, we have two equivalent ways to represent an edge. Let e_x ($e_x = 1, \dots, m$) be the index of the nodes (i.e., the edges in \mathbf{A}) in the line graph. We can also represent the edge e_x by the pair of its source and target nodes in the original graph \mathbf{A} : $\langle i_x, j_x \rangle$, i.e., the edge e_x starts with the node i_x and ends at node j_x .

In order to design an effective strategy to optimize the graph structure to affect the outcome of an information dissemination process, we need to answer the following three questions. (1) (*Key graph parameters/metrics*) What are key graph metrics/parameters that determine/control the dissemination process? (2) (*Graph operations*) What types

Table 8.1: Symbols

Symbol	Definition and Description
$\mathbf{A}, \mathbf{B}, \dots$	matrices (bold upper case)
$\mathbf{A}(i, j)$	the element at the i^{th} row and the j^{th} column of \mathbf{A}
$\mathbf{A}(i, :)$	the i^{th} row of matrix \mathbf{A}
$\mathbf{A}(:, j)$	the j^{th} column of matrix \mathbf{A}
\mathbf{A}'	transpose of matrix \mathbf{A}
$\mathbf{a}, \mathbf{b}, \dots$	vectors
$\mathcal{J}, \mathcal{J}, \dots$	sets (calligraphic)
λ	the largest (in module) eigenvalue of \mathbf{A}
\mathbf{u}, \mathbf{v}	the $n \times 1$ left eigenvector and right eigenvector associated with λ .
n	the number of the nodes in the graph
m	the number of the edges in the graph
k	the budget (i.e., the number of deleted or added edges)

of graph operations (e.g., deleting nodes/edges, adding edges, etc) are we allowed to change the graph structure? (3) (*Affecting algorithms*) For a given graph operation, how can we design effective, scalable algorithms to optimize the corresponding key graph parameters?

For information dissemination on real graphs, we have already seen that for a large family of dissemination processes, the largest (in modulus) eigenvalue λ of the adjacency matrix \mathbf{A} or an appropriately defined system matrix is the *only* graph parameter that determines the tipping point of the dissemination process, i.e., whether or not the dissemination will become an epidemic. In principle, this gives a clear guidance on the algorithmic side, that is, *an ideal, optimal strategy to affect the outcome of the information dissemination process should change the graph structure so that the leading eigenvalue λ is minimized or maximized.*

Based on this observation, now we can transform the original problem of affecting the dissemination process to the **eigenvalue optimization problem**, that is,

- (1) minimize the leading eigenvalue λ for NETMELT;
- (2) maximize the leading eigenvalue λ for NETGEL.

In this chapter, we focus on operating on the *edge-level* to design affecting algorithms. With the above notation, our problems can be formally defined as the following two sub-problems:

Problem 8.1. NETMELT (Edge Deletion)

Given: A large $n \times n$ graph \mathbf{A} and an integer (budget) k ;

Output: A set of k edges from \mathbf{A} whose deletion from \mathbf{A} creates the largest decrease of the leading eigenvalue of \mathbf{A} .

Problem 8.2. NETGEL (Edge Addition)

Given: A large $n \times n$ graph \mathbf{A} and an integer (budget) k ;

Find: A set of k non-edges of \mathbf{A} whose addition to \mathbf{A} creates the largest increase of the leading eigenvalue of \mathbf{A} .

As we will show soon, both problems are combinatorial.

8.3 Proposed Algorithm for NETMELT

In this section, we address the NETMELT problem (Prob. 8.1), that is, to delete k edges from the original graph \mathbf{A} so that its leading eigenvalue λ will decrease as much as possible. We first study the relationship between two different strategies (edge deletion vs. node deletion), and then present our algorithm, followed by the analysis of its effectiveness as well as efficiency.

8.3.1 Edge Deletion vs. Node Deletion

Roughly speaking, in the NETMELT Problem (Edge Deletion), we want to find a set of k ‘important’ edges from the graph \mathbf{A} to delete. With the notation of the line graph $L(\mathbf{A})$, intuitively, such ‘important’ edges in \mathbf{A} might become ‘important’ nodes in the line graph $L(\mathbf{A})$. In this section, we briefly present the relationship between these two strategies (node deletion vs. edge deletion).

Our main result is summarized in Lemma 8.1, which says that the eigenvalues of the original graph \mathbf{A} are also the eigenvalues of its line graph $L(\mathbf{A})$.

Lemma 8.1. Line Graph Spectrum. *Let λ be an eigenvalue of the graph \mathbf{A} . Then λ is also the eigenvalue of the line graph $L(\mathbf{A})$.*

PROOF. Let \mathbf{u} and \mathbf{v} be the left and right eigenvectors of the graph \mathbf{A} that correspond to any eigenvalue λ . We have $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$ and $\mathbf{u}'\mathbf{A} = \lambda\mathbf{u}'$. Equivalently, we have:

$$\lambda\mathbf{v}(i) = \sum_{j:\mathbf{A}(i,j)=1} \mathbf{v}(j) \quad \text{and} \quad \lambda\mathbf{u}(j) = \sum_{i:\mathbf{A}(i,j)=1} \mathbf{u}(i) \quad (8.1)$$

Let e_x ($e_x = 1, \dots, m$) be an edge of the graph \mathbf{A} . Recall that we can also represent e_x by its source and target nodes: $\langle i_x, j_x \rangle$. Let $\tilde{\mathbf{A}}$ be the adjacency matrix of the line graph $L(\mathbf{A})$. By the definition of the line graph, we have $\tilde{\mathbf{A}}(e_x, e_y) = 1$ if $j_x = i_y$, and $\tilde{\mathbf{A}}(e_x, e_y) = 0$ otherwise ($e_x, e_y = 1, \dots, m$).

We define two $m \times 1$ vectors $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$ as: $\tilde{\mathbf{u}}(e_x) = \mathbf{u}(i_x)$ and $\tilde{\mathbf{v}}(e_x) = \mathbf{v}(j_x)$ with $e_x = 1, \dots, m$.

Next, we will show that $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$ are the left and right eigenvectors of $\tilde{\mathbf{A}}$ respectively, with λ being the corresponding eigenvalue.

For any edge e_x ($e_x = 1, \dots, m$), we have

$$\begin{aligned}
\tilde{\mathbf{A}}(e_x, :)\tilde{\mathbf{v}} &= \sum_{e_y: \tilde{\mathbf{A}}(e_x, e_y)=1} \tilde{\mathbf{v}}(e_y) \\
&= \sum_{e_y: i_y=j_x} \mathbf{v}(j_y) \text{ (all edges adjacent from } e_x) \\
&= \sum_{j_y: \mathbf{A}(j_x, j_y)=1} \mathbf{v}(j_y) \text{ (all edges from node } j_x) \\
&= \lambda \mathbf{v}(j_y) \text{ (due to Eq. (8.1))} \\
&= \lambda \tilde{\mathbf{v}}(e_x) \text{ (due to dfn. of } \tilde{\mathbf{v}})
\end{aligned} \tag{8.2}$$

Similarly, for any edge e_x ($e_x = 1, \dots, m$), we have

$$\begin{aligned}
\tilde{\mathbf{u}}'\tilde{\mathbf{A}}(:, e_x) &= \sum_{e_y: \tilde{\mathbf{A}}(e_y, e_x)=1} \tilde{\mathbf{u}}(e_y) \\
&= \sum_{e_y: j_y=i_x} \mathbf{u}(i_y) \text{ (all edges adjacent to } e_x) \\
&= \sum_{j_y: \mathbf{A}(j_y, i_x)=1} \mathbf{u}(i_y) \text{ (all edges to node } i_x) \\
&= \lambda \mathbf{u}(i_x) \text{ (due to Eq. (8.1))} \\
&= \lambda \tilde{\mathbf{u}}(e_x) \text{ (due to dfn. of } \tilde{\mathbf{u}})
\end{aligned} \tag{8.3}$$

Putting Eq. (8.2) and Eq. (8.3) together, we have $\tilde{\mathbf{A}}\tilde{\mathbf{v}} = \lambda\tilde{\mathbf{v}}$ and $\tilde{\mathbf{u}}'\tilde{\mathbf{A}} = \lambda\tilde{\mathbf{u}}'$, which completes the proof. \square

By Lemma 8.1, it seems that edge deletion (Prob. 8.1) can be transformed to the node deletion problem on the line graph – that is, select a subset of k nodes from the line graph $L(\mathbf{A})$ whose deletion creates the largest decrease in terms of the leading eigenvalue of $L(\mathbf{A})$. However, by the following lemma, the node deletion problem itself is still a challenging task.

Lemma 8.2. Hardness of Node Deletion. *It is NP-Complete to find a set of k nodes from a graph \mathbf{A} , whose deletion will create the largest decrease of the largest eigenvalue of the graph \mathbf{A} .*

PROOF. The proof can be done by the reduction from the independent node set problem, which is known to be NP-Complete [Kar72]. The detailed proof is omitted for brevity. \square

That said, we seek an effective algorithm that directly solves the NETMELT problem next.

8.3.2 Proposed K-EDGEDELETION Algorithm

The key to solving Prob. 8.1 (NETMELT) is to quantify the impact of deleting a set of edges in terms of the leading eigenvalue λ . The naive way is to recompute the leading eigenvalue λ after deleting the corresponding set of edges - the smaller the new eigenvalue, the better the subset of the edges. But it is computationally infeasible for large graphs since it takes $O(m)$ time for each of the $\binom{m}{k}$ possible sets, as in general, the impact for a given set of the edges (in terms of decreasing the leading eigenvalue λ) is *not* equal to the summation of the impact of deleting each individual edge.

Let \mathbf{u} and \mathbf{v} be the leading left eigenvector and right eigenvector of the graph \mathbf{A} , respectively. Intuitively, the left eigen-score $\mathbf{u}(i)$ and the right eigen-score $\mathbf{v}(j)$ ($i, j = 1, \dots, n$) provide some importance measure for the corresponding nodes i and j . The core idea of the proposed K-EDGEDELETION algorithm is to quantify the impact of each edge by the corresponding left and right eigen-scores *independently* (step 9). Our upcoming analysis in the next subsection shows that this strategy (1) leads to a good approximation of the actual impact wrt decreasing the leading eigenvalue; and (2) naturally de-couples the dependence among the different edges. As a result, we can avoid the combinatorial enumeration in Prob. 8.1 by picking the top- k edges with the highest individual impact scores (step 9).

Note that steps 2-7 in Alg. 3 are to ensure that all the eigen-scores (i.e., $\mathbf{u}(i), \mathbf{v}(j)$ ($i, j = 1, \dots, n$)) are non-negative. According to the Perron-Frobenius theorem [GL96], such eigenvectors \mathbf{u} and \mathbf{v} always exist.

Algorithm 3 K-EDGEDELETION

Input: the adjacency matrix \mathbf{A} and the budget k

Output: k edges

- 1: compute the leading eigenvalue λ of \mathbf{A} ; let \mathbf{u} and \mathbf{v} be the corresponding left and right eigenvectors, respectively;
 - 2: **if** $\min_{i=1, \dots, n} \mathbf{u}(i) < 0$ **then**
 - 3: assign $\mathbf{u} \leftarrow -\mathbf{u}$
 - 4: **end if**
 - 5: **if** $\min_{i=1, \dots, n} \mathbf{v}(i) < 0$ **then**
 - 6: assign $\mathbf{v} \leftarrow -\mathbf{v}$
 - 7: **end if**
 - 8: **for** each edge $e_x : (i_x, j_x)$ $e_x = 1, \dots, m; i_x, j_x = 1, \dots, n$ **do**
 - 9: score(e_x) = $\mathbf{u}(i_x)\mathbf{v}(j_x)$;
 - 10: **end for**
 - 11: return top- k edges with the highest score(e_x)
-

8.3.3 Proofs and Analysis

Here, we analyze the accuracy and the efficiency of the proposed K-EDGEDELETION algorithm.

The accuracy of the proposed K-EDGEDELETION is summarized in Lemma 8.3. According to Lemma 8.3, the first-order matrix perturbation theory, together with the fact that many real graphs have large eigen-gap, provides a good approximation to the impact of a set of edges in terms of decreasing the leading eigenvalue. What is more important, with such an approximation, the impact of the different edges are now de-coupled from each other. Therefore, we can avoid the combinatorial enumeration of Prob. 8.1 by simply returning the top- k edges with the highest individual impact scores (step 9 in Alg. 3).

Notice that by Lemma 8.3, there is an $O(k)$ gap between the approximate and the actual impact of a set of edges in terms of decreasing the leading eigenvalue. Our experimental evaluations show that the correlation between the approximate and the actual impact is very high (See Section 6 for details), indicating that it indeed provides a good approximation for the actual decrease of the leading eigenvalue.

Lemma 8.3. *Let $\hat{\lambda}$ be the (exact) first eigenvalue of $\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$ is the perturbed version of \mathbf{A} by removing all of its edges indexed by the set \mathcal{S} . Let $\delta = \lambda - \lambda_2$ be the eigen-gap of the matrix \mathbf{A} where λ_2 is the second eigenvalue of \mathbf{A} , and $c = 1/(\mathbf{u}'\mathbf{v})$. If λ is the simple first eigenvalue of \mathbf{A} , and $\delta \geq 2\sqrt{k}$, then $\lambda - \hat{\lambda} = c \sum_{e_x \in \mathcal{S}} \mathbf{u}(i_x)\mathbf{v}(j_x) + O(k)$.*

PROOF. Let $\lambda_i (i = 1, \dots, n)$ be the ordered eigenvalues of \mathbf{A} (i.e., $|\lambda| = |\lambda_1| \geq |\lambda_2| \dots \geq |\lambda_n|$). Let $\tilde{\lambda}_i (i = 1, \dots, n)$ be the corresponding eigenvalues of $\hat{\mathbf{A}}$. Notice that we omitted the subscripts for the leading eigenvalues (i.e., $\lambda_1 = \lambda$, and $\tilde{\lambda}_1 = \tilde{\lambda}$).

Let $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{E}$. We have $\|\mathbf{E}\|_{\text{Fro}} = \sqrt{k}$.

According to the first-order matrix perturbation theory (p.183 [SS90]), we have

$$\begin{aligned} \tilde{\lambda}_1 &= \lambda_1 + \frac{\mathbf{u}'\mathbf{E}\mathbf{v}}{\mathbf{u}'\mathbf{v}} + O(\|\mathbf{E}\|^2) \\ &= \lambda_1 - c \sum_{e_x \in \mathcal{S}} \mathbf{u}(i_x)\mathbf{v}(j_x) + O(k) \end{aligned} \quad (8.4)$$

Next, we will show that $\tilde{\lambda}_1$ is indeed the leading eigenvalue of $\hat{\mathbf{A}}$. To this end, again by the matrix perturbation theory (p.203 [SS90]), we have

$$\begin{aligned} \tilde{\lambda}_1 &\geq \lambda_1 - \|\mathbf{E}\|_2 \geq \lambda_1 - \|\mathbf{E}\|_{\text{Fro}} \geq \lambda_1 - \sqrt{k} \\ \tilde{\lambda}_i &\leq \lambda_i + \|\mathbf{E}\|_2 \leq \lambda_i + \|\mathbf{E}\|_{\text{Fro}} \leq \lambda_i + \sqrt{k} (i \geq 2) \end{aligned} \quad (8.5)$$

Since $\delta = \lambda_1 - \lambda_2 \geq 2\sqrt{k}$, we have $\tilde{\lambda}_1 \geq \tilde{\lambda}_i (i = 2, \dots, n)$. In other words, we have that $\tilde{\lambda}_1 = \hat{\lambda}$ is the leading eigenvalue of $\hat{\mathbf{A}}$. Therefore,

$$\lambda - \hat{\lambda} = c \sum_{e_x \in \mathcal{S}} \mathbf{u}(i_x)\mathbf{v}(j_x) + O(k) \quad (8.6)$$

which completes the proof. \square

The efficiency of the proposed K-EDGEDELETION is summarized in the following lemma, which says that with a fixed budget k , K-EDGEDELETION is *linear* wrt the size of the graph for both time and space cost.

Lemma 8.4. Efficiency of K-EDGEDELETION. *The time cost of Alg. 3 is $O(mk + n)$. The space cost of Alg. 3 is $O(n + m + k)$.*

PROOF. Using the power method, step 1 takes $O(m)$ time. Steps 2-7 take $O(n)$ time. Steps 8-10 take $O(m)$ time. Step 11 takes $O(mk)$ time. Therefore, the overall time complexity of Alg. 3 is $O(mk + n)$, which completes the proof of the time cost.

We need $O(m)$ to store the original graph \mathbf{A} . It takes $O(n)$ and $O(1)$ to store the eigenvectors and eigenvalue, respectively. We need additional $O(m)$ to store the scores (Step 9) for all the edges. Finally, it takes $O(k)$ for the selected k edges. Therefore, the overall space complexity of Alg. 3 is $O(m + n + k)$, which completes the proof of the space cost. \square

8.4 Proposed Algorithm for NETGEL

In this Section, we address the NETGEL problem (Prob. 8.2), where we want to *add* a set of new links into the graph \mathbf{A} so that its leading eigenvalue λ will increase as much as possible. We first present the proposed K-EDGEADDITION algorithm, and then analyze its accuracy as well as efficiency.

8.4.1 Proposed K-EDGEADDITION Algorithm

Let \mathcal{T} be a set of non-existing edges in \mathbf{A} , that is, for each $e_x : \langle i_x, j_x \rangle \in \mathcal{T}$, we have $\mathbf{A}(i_x, j_x) = 0$. Let $\hat{\lambda}$ be the leading eigenvalue of the new adjacency matrix $\hat{\mathbf{A}}$ by introducing the new edges indexed by the set \mathcal{T} . By the similar procedure as in the proof of Lemma 8.3, we can show that the impact of the new set of edges \mathcal{T} in terms of increasing the leading eigenvalue $\hat{\lambda} - \lambda$ can be approximated as

$$\hat{\lambda} - \lambda \approx \sum_{e_x \in \mathcal{T}} \mathbf{u}(i_x) \mathbf{v}(j_x) \quad (8.7)$$

Therefore, it seems that we could use a similar procedure as K-EDGEDELETION to solve the NETGEL problem (referred to as ‘*Naive-Add*’): for each non-existing edge $e_x : \langle i_x, j_x \rangle$, calculate its score as $\text{score}(e_x) = \mathbf{u}(i_x) \mathbf{v}(j_x)$; and pick top- k non-existing edges with the highest scores.

However, many real graphs are very sparse, i.e., $m \ll n^2$. Therefore, we have $O(n^2 - m) \approx O(n^2)$ possible non-existing edges. In other words, *Naive-Add* requires

quasi-quadratic time wrt the number of the nodes (n) in the graph, which does not scale to large graphs.

To address this issue, we propose an efficient algorithm, which is summarized in Alg 4. The core idea of K-EDGEADDITION is to prune a large portion of the non-existing edge pairs based on their left and right eigen-scores. As in Alg. 3, we take the same procedure to make sure that the left and right eigenvectors (\mathbf{u}, \mathbf{v}) are non-negative. We omit these steps in Alg 4 for brevity.

Algorithm 4 K-EDGEADDITION

Input: the adjacency matrix \mathbf{A} and the budget k

Output: k non-existing edges

- 1: compute the left (\mathbf{u}) and right (\mathbf{v}) eigenvectors of \mathbf{A} that correspond to the leading eigenvalue ($\mathbf{u}, \mathbf{v} \geq 0$);
 - 2: calculate the maximum in-degree (d_{in}) and out-degree (d_{out}) of \mathbf{A} , respectively;
 - 3: find the subset of $k + d_{in}$ nodes with the highest left eigen-scores \mathbf{u}_i . Index them by \mathcal{I} ;
 - 4: find the subset of $k + d_{out}$ nodes with the highest right eigen-scores \mathbf{v}_j . Index them by \mathcal{J} ;
 - 5: **for** each edge $e_x : \langle i_x, j_x \rangle \ i_x \in \mathcal{I}, j_x \in \mathcal{J}, \mathbf{A}(i_x, j_x) = 0$ **do**
 - 6: score(e_x) = $\mathbf{u}(i_x)\mathbf{v}(j_x)$. Index them by \mathcal{P} ;
 - 7: **end for**
 - 8: return top- k non-existing edges with the highest scores among \mathcal{P} .
-

8.4.2 Proofs and Analysis

Here, we analyze the accuracy and efficiency of the proposed K-EDGEADDITION.

The accuracy of the proposed K-EDGEADDITION is summarized in Lemma 8.5, which says that K-EDGEADDITION selects the same set of edges as *Naive-Add*.

Lemma 8.5. Effectiveness of K-EDGEADDITION. *Alg. 4 outputs the same set of non-existing edges as Naive-Add.*

PROOF. Omitted for brevity. □

The efficiency of the proposed K-EDGEADDITION is summarized in the following lemma.

Lemma 8.6. Efficiency of K-EDGEADDITION. *The time cost of Alg. 4 is $O(m + nt + kt^2)$. The space cost of Alg. 4 is $O(n + m + t^2)$, where $t = \max(k, d_{in}, d_{out})$.*

PROOF: Using the power method, step 1 takes $O(m)$ time. Step 2 takes $O(m + n)$ time. Steps 3-4 take $O(n(d_{in} + k))$ and $O(n(d_{out} + k))$ time respectively, both of which can be written as $O(nt)$. Steps 5-7 take $O((k + d_{in})(k + d_{out})) = O(t^2)$ time. Step 8 takes

$O((k + d_{in})(k + d_{out})k) = O(kt^2)$. Therefore, the overall time cost is $O(m + nt + kt^2)$, which completes the proof of the time complexity.

We need $O(m)$ to store the original graph \mathbf{A} . It takes $O(n)$ to store the eigenvectors \mathbf{u} and \mathbf{v} . Step 2 takes additional $O(n + 1)$ space. Steps 3-4 take $O(d_{in} + k)$ and $O(d_{out} + k)$ space respectively, both of which can be simplified as $O(t)$. Steps 5-7 take at most $O((k + d_{in})(k + d_{out})) = O(t^2)$ space. Step 9 takes $O(k)$ space. Therefore, the overall space cost (by omitting the smaller terms) is $O(m + nt + kt^2)$, which completes the proof of the space complexity. \square

8.5 Experimental Evaluations

Dataset	n	m
<i>Oregon-A</i>	633	2,172
<i>Oregon-B</i>	1,503	5,620
<i>Oregon-C</i>	2,504	9,446
<i>Oregon-D</i>	2,854	9,864
<i>Oregon-E</i>	3,995	15,420
<i>Oregon-F</i>	5,296	20,194
<i>Oregon-G</i>	7,352	31,330
<i>Oregon-H</i>	10,860	46,818
<i>Oregon-I</i>	13,947	61,168

Table 8.2: Dataset summary.

Dataset	k = 10	k = 50	k = 100	k = 500	k = 1000
<i>Oregon-A</i>	0.999	0.997	0.995	0.973	0.924
<i>Oregon-B</i>	0.999	0.999	0.998	0.993	0.988
<i>Oregon-C</i>	1.000	0.999	0.999	0.996	0.991
<i>Oregon-D</i>	0.999	0.999	0.999	0.994	0.988
<i>Oregon-E</i>	1.000	0.999	0.999	0.998	0.995
<i>Oregon-F</i>	1.000	0.999	0.999	0.998	0.997
<i>Oregon-G</i>	1.000	0.999	0.999	0.999	0.998
<i>Oregon-H</i>	1.000	1.000	0.999	0.999	0.999
<i>Oregon-I</i>	1.000	1.000	0.999	0.999	0.999

Table 8.3: Evaluations on the approx. quality. Larger is better.

In this section, we provide empirical evaluations for the proposed K-EDGEDELETION and K-EDGEADDITION algorithms. Our evaluations mainly focus on (1) the effectiveness and (2) the efficiency of the proposed algorithms.

8.5.1 Experimental Setup

Data sets. We used a popular set of real graphs for our experiments - the Oregon AS (Autonomous System) router graphs, which are AS-level connectivity networks inferred from Oregon route-views². These were collected once a week, for 9 consecutive weeks. Table 8.2 summarizes the nine graphs we used in our evaluations.

Evaluation criteria. As mentioned before, the leading eigenvalue λ of the graph is the only graph parameter that determines the epidemic threshold for a large family of information dissemination processes. Therefore, we report the change of the leading eigenvalue for the effectiveness comparison - for both NETMELT and NETGEL problems. A larger change of the leading eigenvalue is better, which suggests that we can affect the outcome of the dissemination process more. In addition, we also run virus propagation simulations to compare how different methods affect the actual outcome of the propagation. For the computational cost and scalability, we report the wall-clock time.

Machine configurations. All the experiments ran on the same machine with four 2.4GHz AMD CPUs and 48GB memory, running Linux (2.6 kernel).

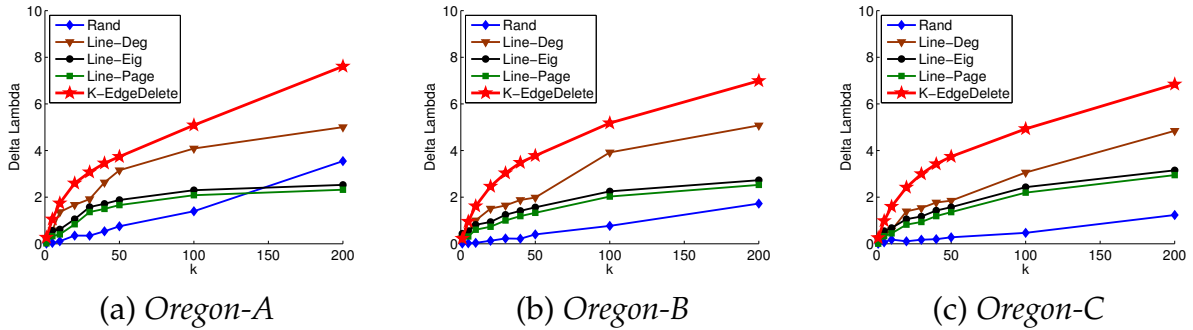


Figure 8.2: The decrease of the leading eigenvalue vs. the budget k . Larger is better. The proposed K-EDGEDELETION always leads to the biggest decrease of the leading eigenvalue.

8.5.2 Effectiveness of K-EDGEDELETION

Approximation Quality. For both K-EDGEDELETION and K-EDGEADDITION, we want to approximate the actual change of the leading eigenvalue by the first order matrix perturbation theory. This is the *only* place we introduce the approximation. By Lemma 8.3,

²<http://topology.eecs.umich.edu/data.html>

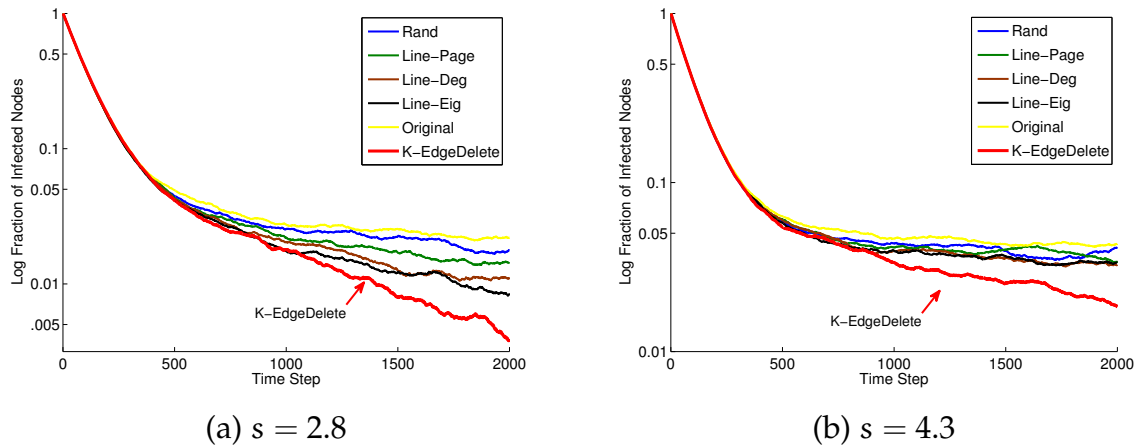


Figure 8.3: Comparison of minimizing the outcome of the virus propagation. Fraction of infected nodes vs. time stamp. Lower is better. The proposed K-EDGEDELETION always leads to the least number of infected nodes. Notice that y-axis is in the logarithmic scale.

it says that the quality of such an approximation depends on both the budget k as well as the eigengap of the original graph, with an $O(k)$ gap. Here, let us experimentally evaluate how good this approximation is on real graphs. We compute the linear correlation coefficient between the actual and approximate leading eigenvalue after we randomly remove k ($k = 10, 50, 100, 500, 1000$) edges. The results are shown in table 8.3. It can be seen that the approximation is very good - in all the cases, the linear correlation coefficient is greater than 0.92, and often it is very close to 1.

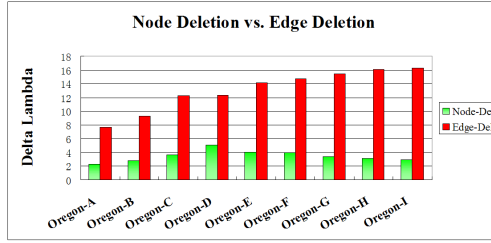
The Impact of Decreasing the Leading Eigenvalue. Here, we evaluate the effectiveness of the proposed K-EDGEDELETION in terms of decreasing the leading eigenvalue λ of the graph. Lemma 8.1 suggests that the ‘important’ edges on the original graph \mathbf{A} might become ‘important’ nodes on the line graph $L(\mathbf{A})$. We follow this intuition to design the following comparative strategies: (1) randomly select k edges from the original graph \mathbf{A} (referred to as ‘Rand’); (2) select k edges with the highest degrees in the line graph $L(\mathbf{A})$ (referred to as ‘Line-Deg’); (3) select k edges with the highest eigen-scores in the line graph $L(\mathbf{A})$ (referred to as ‘Line-Eig’); and (4) select k edges with the highest PageRank scores in the line graph $L(\mathbf{A})$ (referred to as ‘Line-Page’). For ‘Rand’, we run the experiments 100 times and report the average result. For ‘Line-Deg’, we have two variants by using out-degree or in-degree. In our evaluation, we found that these two variants give the similar results. Therefore, we only report the results by out-degree. For the same reason, we only report the results by the right eigen-scores for ‘Line-Eigs’. For ‘Line-Page’, there is an additional parameter of the teleport probability. We run the experiments with the different teleport probabilities and report the best results.

For brevity, we only present the results on *Oregon-A*, *Oregon-B* and *Oregon-C* since the results on the rest six graphs are similar. From Fig. 8.2, it can be seen that our K-

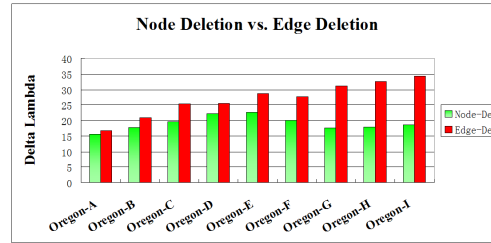
EDGEDELETION always leads to the biggest decrease in terms of the leading eigenvalue. For example, on *Oregon-C* graph, the proposed K-EDGEDELETION decreases the leading eigenvalue by 3.8 with the budget $k = 50$, which is almost double of the second best method (e.g., 2.0 by ‘Line-Deg’). Therefore, we expect that K-EDGEDELETION would affect the outcome of the dissemination processes better than the alternative choices, e.g., having less number of infected nodes in the graph, etc. We validate this next.

Affecting Virus Propagation. Next, we evaluate the effectiveness of the proposed K-EDGEDELETION in terms of minimizing the outcome of the information dissemination processes. To this end, we simulate the virus propagation for the SIS model (susceptible-infective-susceptible) on the graph [WCWF03]. For each method, we delete $k = 200$ edges from the original graph. Let $s = \lambda b/d$ be the normalized virus strength (bigger s means stronger virus), where b and d are the infection rate and death rate, respectively. The results are presented in Fig. 8.3, which is averaged over 1,000 runs. It can be seen that the proposed K-EDGEDELETION is always the best - its curve is always the lowest which means that we always, as desired, have the least number of infected nodes in the graph with this strategy. In Fig. 8.3, ‘Original’ (the yellow curve) means that we simulate the virus propagation on the *original* graph without deleting any edges. Notice that when the virus becomes stronger (Fig. 8.3(b)), all the curves except the proposed method mix with ‘Original’, which means that they all fail to affect the virus propagation in this case. In contrast, our proposed method (the red curve) can still significantly reduce the number of infected nodes.

Node Deletion vs. Edge Deletion. Finally, in some applications, e.g., to stop malware propagation on the computer networks, both node deletion (e.g., shutting down some machines) and edge deletion (e.g., blocking some links between machines) are feasible. In this case, we want to know which strategy (node deletion or edge deletion) is more effective in affecting the outcome of such propagation process. To this end, we use an effective node immunization algorithm [TPT⁺10] to delete $\tilde{k} = 1, 10$ nodes respectively (referred to as ‘Node-Del’). For each \tilde{k} , we then use our proposed K-EDGEDELETION to delete the same amount of edges from the original graph (referred to as ‘Edge-Del’). We compare the decrease of the leading eigenvalues of the two methods. The results are summarized in Fig. 8.4. It can be seen that ‘Edge-Del’ always leads to a bigger decrease of the leading eigenvalue - which suggests that by operating on the edge level, we can design a more effective algorithm with the same budget to affect the outcome of the information dissemination process. The results are consistent with the intuition - not all the edges adjacent to the ‘important’ nodes, which the node immunization algorithm aims to delete, are also ‘important’ (e.g., many edges adjacent to an ‘important’ node might link to/from some degree-1 nodes). In other words, edge deletion enables us to optimize the underlying graph structure on a finer granularity by picking each individual edge one by one.



(a) $\tilde{k} = 1$



(b) $\tilde{k} = 10$

Figure 8.4: Comparison between node deletion vs. edge deletion. Larger is better. With the same amount of edges deleted, our proposed K-EDGEDELETION (red) leads to a bigger decrease in terms of the leading eigenvalue.

8.5.3 Effectiveness of K-EDGEADDITION

To our best knowledge, there are no existing methods to add k new links into an existing graph in order to increase its leading eigenvalue. Let $\bar{\mathbf{A}}$ be the complementary graph of \mathbf{A} , which has the same node set as \mathbf{A} , and $\bar{\mathbf{A}}(i, j) = 1$ iff $\mathbf{A}(i, j) = 0$. With the notation of the complementary graph, we use the following intuition to design the comparative methods: to select k ‘important’ edges from the *complementary graph* $\bar{\mathbf{A}}$ and add them into the original graph \mathbf{A} . More specifically, we compare the proposed K-EDGEADDITION with the following strategies: (1) randomly select k edges (referred to as ‘Rand’); (2) select k edges with the highest out-degrees in the line graph of the complementary graph $\bar{\mathbf{A}}$ (referred to as ‘CompDeg’); (3) select k edges with the highest right eigen-scores in the line graph of the complementary graph $\bar{\mathbf{A}}$ (referred to as ‘CompEigs’); (4) select k edges with the highest PageRank scores in the line graph of the complementary graph $\bar{\mathbf{A}}$ (referred to as ‘CompPage’); and (5) select k edges by running K-EDGEDELETION in the complementary graph $\bar{\mathbf{A}}$ (referred to as ‘CompDelete’). Again, for ‘Rand’, we run the experiments 100 times and report the average result. We only report the results of ‘CompDeg’ by out-degree and those of ‘CompEig’ by right eigen-scores, respectively, since the other variants give the similar performance. For ‘CompPage’, we run the experiments with the different teleport probabilities and report the best results.

The Impact of Increasing the Leading Eigenvalue. We first evaluate the effectiveness of the proposed K-EDGEADDITION in terms of *increasing* the leading eigenvalue of the graph. For brevity, we only present the results on *Oregon-A*, *Oregon-B* and *Oregon-C* since the results on the rest of the graphs are similar. From Fig. 8.5, it can be seen that the proposed K-EDGEADDITION always leads to the biggest increase in terms of the leading eigenvalue of the graph. Notice that for all the comparative methods, they behave like ‘Rand’ (blue curve), especially when the budget k is small.

Affecting Virus Propagation. We also evaluated the effectiveness of the proposed K-EDGEADDITION in terms of *maximizing* the outcome of the information dissemination process. To this end, again, we simulate the virus propagation for the SIS model on the graph. For each method, we add $k = 200$ new edges into the graph. Again, let $s = \lambda b/d$ be the normalized virus strength, with bigger s being stronger virus. Here, our goal is to *increase* the number of ‘infected’ nodes (e.g., having more people in the social networks to adopt a piece of good idea, etc) by introducing a set of new links into the graph. The result is presented in Fig. 8.6, which is averaged over 1,000 runs. It can be seen that the proposed K-EDGEADDITION is always the best - its curve is always the highest which means that we always have the largest number of ‘infected’ nodes in the graph with this strategy. Notice that when the strength of the virus is weak (Fig. 8.6(a)), all the curves except the proposed method mix with or are very close to ‘Original’ (yellow curve), which means that they have little impact to boost the outcome of the propagation in this case. In contrast, our proposed method (the red curve) can still significantly increase the number of ‘infected’ nodes. Therefore, we conclude that our proposed K-EDGEADDITION is much more effective to guild the outcome of the dissemination process.

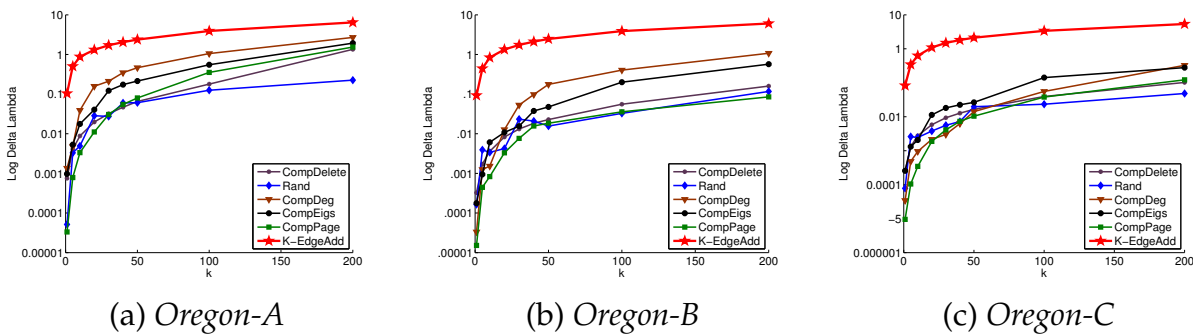


Figure 8.5: The increase of the leading eigenvalue vs. the budget k . Larger is better. The proposed K-EDGEADDITION always leads to the largest increase of the leading eigenvalue. Notice that y-axis is in the logarithmic scale.

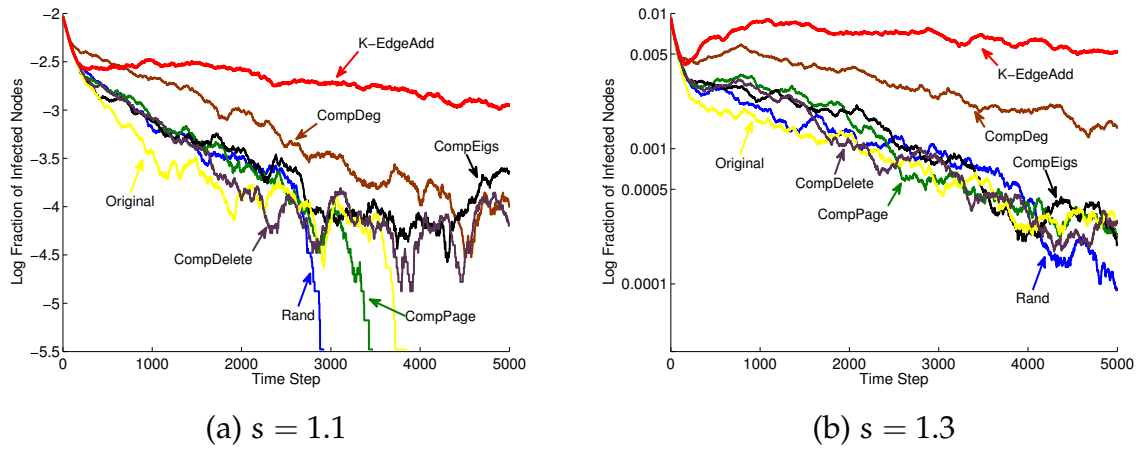


Figure 8.6: Comparison of maximizing the outcome of virus propagation. Fraction of ‘infected’ nodes vs. time stamp. Larger is better. The proposed K-EDGEADDITION always leads to the largest number of ‘infected’ nodes. Notice that y-axis is in the logarithmic scale.

8.5.4 Scalability

We use the subsets of the largest data set *Oregon-I* to evaluate the scalability of the proposed algorithms. The results are presented in Fig. 8.7. We can see that the proposed K-EDGEDELETION and K-EDGEADDITION scale almost near-linearly wrt m , which means that they are suitable for large graphs. Notice that for both cases, we also observe a slight super-linear trend. This is due to the following two reasons: (1) for both K-EDGEDELETION and K-EDGEADDITION, we use the power method to compute the leading eigenvalue and the corresponding eigenvectors. When m increases, the actually iteration number in the power method also tends to increase; (2) for K-EDGEADDITION when m increases, the maximum degree ($\max(d_{in}, d_{out})$) also increases even though we fix the number of the nodes (n).

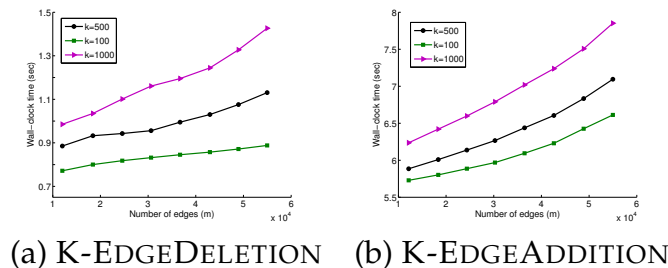


Figure 8.7: Scalability of proposed algorithms. Both K-EDGEDELETION and K-EDGEADDITION scale near-linearly wrt the size of the graph.

8.6 Related Work

In this section, we review the related work specific to the problems discussed in this chapter. We have already reviewed most of the related work in previous chapters (more specifically see Chapters 6 and 7).

Affecting Algorithms. Note that all the previous works focus on operating on the node level (i.e., delete or inoculate a set of ‘best’ nodes) to affect the outcome of the dissemination. In contrast, we study the equally important, but much less studied affecting algorithms by operating on the edge level.

There exist some *empirical evaluations* on edge removal strategies for slightly different purposes, such as, slowing down the influenza spreading [MK09], minimizing the average infection probability [SMHH11], evaluating and comparing the attack vulnerability [HKYH02], etc. The closest related work to our K-EDGEDELETION algorithm is [BS11], which proposed a convex optimization based approach to approximately minimize the leading eigenvalue of the graph. However, the method is based on semi-definite programming and does not scale to large graphs. Moreover, for all these methods, it remains unclear if they can be generalized to address the even more challenging NETGEL problem, where we want to *add* new edges to promote the information dissemination.

Measuring the Importance of Nodes and Edges. In the literature, there are a lot of node importance measurements, including betweenness centrality, both the one based on the shortest path [Fre77] and the one based on random walks [New05b, KPST11] PageRank [PBMW98], HITS [Kle98], and coreness score [MW03]. Our work is also related to the so-called k-vital edges problem, which aims to delete a set of links from the graphs to increase the shortest path length [LSH00] or the weight of the minimum spanning tree of the remaining graph [She95]. K-vital edge problem itself is known to be NP-Hard. Other remotely related work includes graph augmentation [PBG11], graph sparsification [KMST10], network inhibition [Phi93] and network-interdiction [Woo93, IW02]. Both network inhibition and network interdiction are NP-Hard.

8.7 Conclusion

In this chapter, we studied the problem of how to optimize the link structure to affect the outcome of information dissemination processes. The main contributions of the work are:

- *Algorithms.* We observe that for a large family of information dissimulation processes, the problem boils down to the eigenvalue optimization problem. We propose an effective, scalable algorithm to optimize such a key graph parameter (i.e., the leading eigenvalue) that controls the information dissemination process, for both NETMELT and NETGEL, respectively;
- *Proofs and Analysis.* We show the *accuracy* (Lemma 3 and Lemma 5) and the *complexity* of our methods (Lemma 4 and Lemma 6); the *hardness* of the problem

(Lemma 2), and the *equivalence* between the different strategies (Lemma 1, Lemma 7 and Lemma 8);

- *Experimental Evaluations.* Our evaluations on real large graphs show that (a) compared with alternative choices to optimize the link structure, our methods are much more effective to affect the outcome of the dissemination process; (b) compared with the node deletion strategy, our K-EDGEDELETION offers a more effective way by operating on the edge level; and (c) both K-EDGEDELETION and K-EDGEADDITION scale to large graphs.

APPENDIX

Higher-Order NETMELT. From Lemma 8.3, it can be seen that the only place we introduce the approximation in Alg. 3 is to approximate the actual decrease of the leading eigenvalue by the first-order matrix perturbation theory. The readers might wonder if we can further improve the quality by using higher-order matrix perturbation theory, while maintaining the linear scalability of the algorithm.

We explored second-order matrix perturbation theory to approximate the actual decrease of the leading eigenvalue, and found that (1) it generates very similar results as the proposed K-EDGEDELETION algorithm and (2) it requires 5-10x more wall-clock time. The reason might be that for the NETMELT problem, the first-order perturbation already gives a very good approximation. Therefore, in practice, we recommend K-EDGEDELETION for simplicity.

Nonetheless, the new algorithm based on the second-order perturbation exhibits some interesting theoretic properties. It also helps understand the relationship between edge deletion and node deletion on the algorithmic level. We present it here for the completeness.

Let $c = \frac{1}{\mathbf{u}^T \mathbf{v}}$, with second-order matrix perturbation, we can approximate³ the impact of deleting a set of edges \mathcal{S} in terms of the leading eigenvalue as:

$$\lambda - \hat{\lambda} \simeq \text{Impact}(\mathcal{S}) = c \left(\sum_{e_x \in \mathcal{S}} \mathbf{u}(i_x) \mathbf{v}(j_x) - \frac{1}{2\lambda} \sum_{e_x \in \mathcal{S}, e_y \in \mathcal{S}, j_x = i_y} \mathbf{u}(i_x) \mathbf{v}(j_y) \right) \quad (8.8)$$

Compared with the first-order perturbation (eq. (8.6)), we have an additional penalized term in eq. (8.8): $\mathbf{u}(i_x) \mathbf{v}(j_y)$ for any two adjacent edges e_x and e_y . The intuition is to encourage the edges in the set \mathcal{S} to be far away (not adjacent) from each other.

By eq. (8.8), the impact of different edges in the set \mathcal{S} is no longer independent with each other. At the first glance, this might complicate the algorithm since now we need to

³These formulae is similar as the one in [MSN10]

optimize at the set level, that is, to find a set of edges that *collectively* maximize eq. (8.8). However, by the following lemma, the impact defined in eq. (8.8) exhibits some nice diminishing return properties.

Lemma 8.7. Second-Order Approximation Properties. *The $\text{Impact}(\mathcal{S})$ defined in eq. (8.8) has the following properties:*

- (1) $\text{Impact}(\Phi) = 0$, where Φ is an empty set;
- (2) $\text{Impact}(\mathcal{S})$ is monotonically non-decreasing wrt the set \mathcal{S} ;
- (3) $\text{Impact}(\mathcal{S})$ is sub-modular wrt the set \mathcal{S} .

PROOF. Easy to check. □

Thanks to such diminishing return properties, it naturally leads to the following greedy algorithm (K-EDGEDELETION++) to find a *near-optimal* subset of edges to delete from the original graph \mathbf{A} . And it can be shown that the overall time complexity of K-EDGEDELETION++ remains linear wrt the size of the graph.

Algorithm 5 K-EDGEDELETION++

Input: the adjacency matrix \mathbf{A} and the budget k

Output: k edges indexed by set \mathcal{S}

- 1: compute the first eigen-value λ of \mathbf{A} ; compute the corresponding left and right eigenvectors \mathbf{u} and \mathbf{v} ($\mathbf{u}, \mathbf{v} \geq 0$), respectively;
 - 2: initialize the set \mathcal{S} to be empty;
 - 3: $\text{score}(e_x) = \mathbf{u}(i_x)\mathbf{v}(j_x)$ ($e_x : \langle i_x, j_x \rangle$, $e_x = 1, \dots, m$);
 - 4: **for** $k_0 = 1, \dots, k$ **do**
 - 5: find $e_0 = \text{argmax}_{e_x, e_x \notin \mathcal{S}} \text{score}(e_x)$;
 - 6: add the new edge $e_0 : \langle i_0, j_0 \rangle$ into \mathcal{S} ;
 - 7: **for each** edge $e_y : \langle i_y, j_y \rangle$ s.t. $j_y = i_0$ **do**
 - 8: $\text{score}(e_y) \leftarrow \text{score}(e_y) - 1/(2\lambda)\mathbf{u}(i_y)\mathbf{v}(j_0)$;
 - 9: **end for**
 - 10: **for each** edge $e_y : \langle i_y, j_y \rangle$ s.t. $i_y = j_0$ **do**
 - 11: $\text{score}(e_y) \leftarrow \text{score}(e_y) - 1/(2\lambda)\mathbf{u}(i_0)\mathbf{v}(j_y)$;
 - 12: **end for**
 - 13: **end for**
-

An interesting property of Alg. 5 is that it builds the equivalence between edge deletion and node deletion on the algorithmic level:

Lemma 8.8. Equivalence of Alg. 5 to Node Immunization. *Let \mathcal{S} be the set of edges by running Alg. 5 on graph \mathbf{A} ; \mathcal{T} be the set of edges by running the node immunization algorithm [TPT⁺10] on the line graph $L(\mathbf{A})$; and $|\mathcal{S}| = |\mathcal{T}|$. We have $\mathcal{S} = \mathcal{T}$.*

Chapter 9

Finding Culprits

Given a snapshot of a large graph, in which an infection has been spreading for some time, can we identify those nodes from which the infection started to spread? In other words, can we reliably tell who the culprits are? In this chapter, we answer this question affirmatively, and give an efficient method called NETSLEUTH for the well-known Susceptible-Infected virus propagation model.

Essentially, we are after that set of seed nodes that best explain the given snapshot. We propose to employ the Minimum Description Length principle to identify the best set of seed nodes and virus propagation ripple, as the one by which we can most succinctly describe the infected graph. We give an highly efficient algorithm to identify likely sets of seed nodes given a snapshot. Then, given these seed nodes, we show we can optimize the virus propagation ripple in a principled way by maximizing likelihood. With all three combined, NETSLEUTH can automatically identify the correct number of seed nodes, as well as which nodes are the culprits.

Experimentation on our method shows high accuracy in the detection of seed nodes, in addition to the correct automatic identification of their number. Moreover, we show NETSLEUTH scales linearly in the number of nodes of the graph.

9.1 Introduction

We focus on a different and difficult question in this chapter: Given a single snapshot of a partly infected network, how we can reliably identify those nodes from which the epidemic started; whether for inoculation to prevent future epidemics, or for exploitation for viral marketing.

As such, given a snapshot of a large graph $G(\mathcal{V}, \mathcal{E})$ in which a subset of nodes $\mathcal{V}' \subseteq \mathcal{V}$ is currently infected, and assuming the Susceptible-Infected (SI) propagation model, we consider the problem of how to efficiently and reliably find those seed nodes $\mathcal{S} \subseteq \mathcal{V}'$ from which the epidemic started, without requiring the user to choose the number seed nodes in advance. In other words, we address the questions: *How many culprits are there, and who are they?*

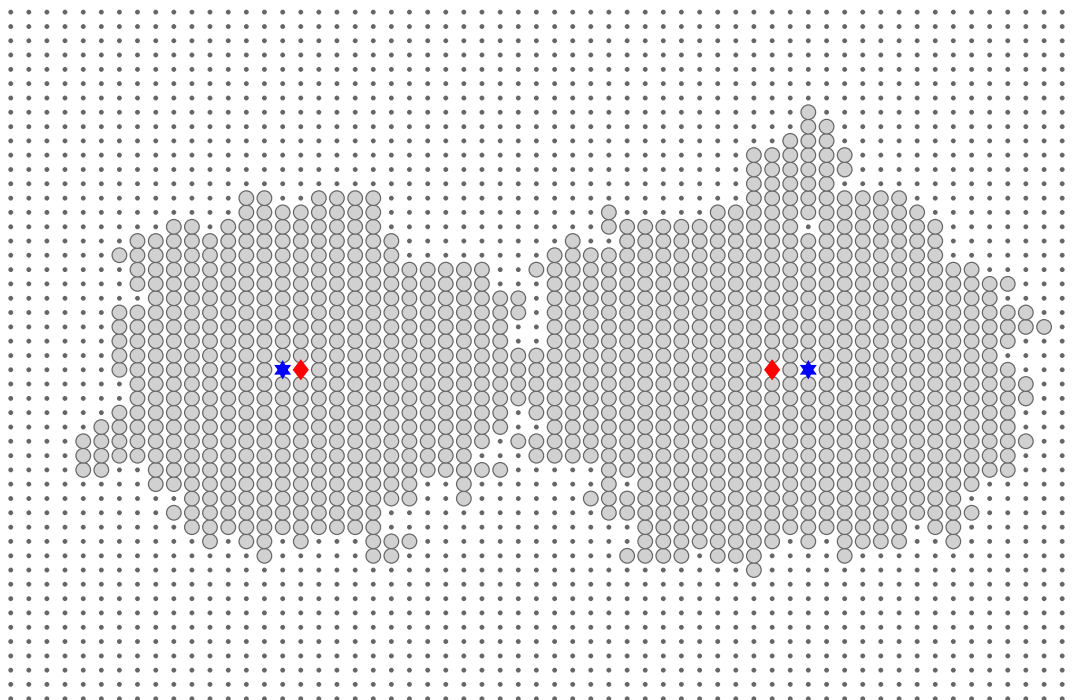


Figure 9.1: Example: Culprits, *how many*, and *which ones*? A snapshot of a 2D grid in which an infection has been stochastically spreading. Grey circles are infected nodes, while Grey dots are un-infected. The 2 Blue stars denote the true seeds. The 2 Red diamonds denote the seeds automatically discovered by NETSLEUTH—that is, both in *number* (two) and *location* (being spatially very close to the true seeds).

We propose to employ the Minimum Description Length (MDL) principle [Gr7] to identify that set of seed nodes, and that virus propagation ripple starting from those nodes that best describes the given snapshot. We give an highly efficient algorithm to identify likely seed nodes, and show we can easily optimize the description length of the virus propagation ripple for a given seed set by greedily maximizing likelihood. As such, we can identify the best set of seed nodes in a principled manner, without having to choose k , the number of seed nodes in advance.

As an example, consider Figure 9.1. It depicts an example grid-structured graph, in which a subgraph has been infected by a stochastic process starting from two seed nodes. The plot shows the true seed nodes, as well as the seed nodes automatically identified by NETSLEUTH; it finds the correct number of seed nodes, and places these where a human would; in fact, the discovered seeds have a higher likelihood for generating this infected subgraph than the true seed nodes.

We develop a two step approach by first finding high-quality seeds given the number of seeds, and then using our carefully designed MDL score to pinpoint the true number of seeds. For the first part, we use the notion of ‘exoneration’ from the un-infected frontier—e.g., in Figure 9.1 the nodes on the edge of the infected snapshot are unlikely to

Table 9.1: Comparison between three culprit-identifying methods: NETSLEUTH, Rumor-centrality [SZ11], and Effectors [LTGM10]

	infection model	$k > 1$	automatically determines k	$O(\cdot)^\dagger$
NETSLEUTH (our method)	SI	✓	✓	Linear
Rumor-centrality [SZ11]	SI	–	–	Quadratic
Effectors [LTGM10]	IC	✓	–	Quadratic

[†] Running time given for arbitrary graphs.

be the culprits due to the large number of un-infected nodes surrounding them. Based on this idea, we develop a novel ‘submatrix-laplacian’ method to find out the best seed sets given a number of seeds (see Section 9.4 for more details). Given these seed-sets, we also give an efficient algorithm to compute the MDL scores, thus finding the number of seeds in a parameter-free way.

Although network infection models have been researched extensively, identifying the seed nodes of an epidemic is surprisingly understudied. We are, however, not the first to research this problem. Recently, Shah and Zaman [SZ10, SZ11] developed rumor-centrality for identifying the *single* source node of an epidemic. In contrast, we allow for *multiple* seed nodes, and automatically determine their number. Lappas et al. [LTGM10] studied the ‘Effectors’ problem of identifying k seed nodes in a steady-state network snapshot, under the Independent Cascade (IC) model. In contrast, we study the SI model, allow the snapshots from any time during the epidemic, and our approach is parameter-free as by MDL we can automatically identify the best value for k . Furthermore, and very importantly for large graphs, in comparison our method is computationally much more efficient. Table 9.1 gives a comparison of NETSLEUTH to these methods. We discuss related work in more detail in Section 9.6.

Experimentation shows that NETSLEUTH detects seed nodes and automatically identifies their number, both with high-accuracy. With synthetic data we show it can handle difficult fringe cases, and is in agreement with human intuition. We show we reliably identify the correct number of seed nodes on real data, and also that our detected seeds are of very high quality (measured by multiple metrics). Finally, we show our method scales linearly with the number of edges of the graph.

The rest of the chapter is organized in the typical way: preliminaries, our problem formulation and method, experiments, related work and then conclusion.

9.2 Preliminaries

In this section we give notation, and introduce MDL and the infection spreading model we use.

9.2.1 Notation

Table 9.2 gives some of the notation and symbols we will be using in this work. We consider undirected, unweighted graphs $G = (\mathcal{V}, \mathcal{E})$ of $N = |\mathcal{V}|$ nodes. All logarithms in this chapter are to base 2, and we adopt the standard convention that $0 \log 0 = 0$. We denote the transpose of any matrix or vector V as V^T . Finally note that L_Λ is a *submatrix* of $L(G)$, *not* the laplacian matrix of G_I .

Table 9.2: Terms and Symbols

Symbol	Definition and Description
SI model	Susceptible-Infected model
β	attack probability of the virus in the SI model
$G = (\mathcal{V}, \mathcal{E})$	graph under consideration
$G_I = (\mathcal{V}_I, \mathcal{E}_I)$	given infected subgraph of G
\mathcal{R}	ripple, list of sets of nodes how virus propagates
N	$ \mathcal{V} $, number of nodes in graph G
N_I	$ \mathcal{V}_I $, number of nodes in graph G_I
$d(i)$	degree of node i
\mathcal{F}	set of un-infected nodes having at least one infected neighbor (in \mathcal{V}_I)
\mathcal{E}_F	set of edges connecting nodes in \mathcal{F} to \mathcal{V}_I
$A(G)$	adjacency matrix of graph G (size $N \times N$)
A	adjacency matrix of G_I (size $N_I \times N_I$)
$D(G)$	diagonal degree matrix of graph G
$L(G)$	laplacian matrix of G i.e. $L(G) = D(G) - A(G)$
L_Λ	<i>submatrix</i> (size $N_I \times N_I$) of $L(G)$ corresponding to the infected graph G_I
Q_{MDL}	MDL-based culprits quality measure (see § 9.5)
Q_{JD}	set-Jaccard-distance-based culprits quality measure (see § 9.5)

9.2.2 The Susceptible-Infected Model

The most basic epidemic model is the so-called ‘Susceptible-Infected’ (SI) model [AM91]. Each object/node in the underlying graph is in one of two states - Susceptible (S) or Infected (I). Once infected, each node stays infected forever. Each infected node tries to infect each of its neighbors independently with probability β in each discrete time-step, which reflects the strength of the virus.

Note that here $1/\beta$ defines a natural time-scale (intuitively it is the expected number of time-steps for a successful attack over an edge). As an example, if we assume that the underlying network is a clique of N nodes, under continuous time, the model can be written as: $\frac{dI(t)}{dt} = \beta(N - I(t))I(t)$, where $I(t)$ is the number of infected nodes at time t —the solution is the logistic function and it is invariant to $\beta \times t$.

9.2.3 Minimum Description Length Principle

The Minimum Description Length principle (MDL) [Gr7], is a practical version of Kolmogorov Complexity [LV93]. Both embrace the slogan *Induction by Compression*. For MDL, this can be roughly described as follows.

Given a set of models \mathcal{M} , the best model $M \in \mathcal{M}$ is the one that minimizes $\mathcal{L}(M) + \mathcal{L}(\mathcal{D} | M)$, in which $\mathcal{L}(M)$ is the length in bits of the description of M , and $\mathcal{L}(\mathcal{D} | M)$ is the length of the description of the data encoded with M .

This is called two-part MDL, or *crude* MDL—as opposed to *refined* MDL, where model and data are encoded together [Gr7]. We use two-part MDL because we are specifically interested in the model: the seed nodes and ripple that give the best description. Further, although refined MDL has stronger theoretical foundations, it cannot be computed except for some special cases. Note that MDL requires the compression to be *lossless* in order to allow for fair comparison between different $M \in \mathcal{M}$.

To use MDL, we have to define what our models \mathcal{M} are, how a $M \in \mathcal{M}$ describes the data at hand, and how we encode this all in bits. Note, that in MDL we are only concerned with code lengths, not actual code words.

9.3 Our Problem Formulation

Next we formulate our problem in terms of MDL. Our goal is to obtain the most succinct explanation of ‘what happened’. To do so, we require two ingredients: the first is a formal objective—a cost function—which we discuss in this section. The second is then an algorithm to find good solutions, which we give in Section 9.4.

Our cost function will consist of two parts, 1) scoring the seed set (Model cost) and 2) scoring the successive infected nodes starting from the seed (Data cost).

We assume that both sender and receiver know the layout of $G = (\mathcal{V}, \mathcal{E})$, but not which nodes are in $G_I = (\mathcal{V}_I, \mathcal{E}_I)$. This makes, using the general formulation of MDL in Section 9.2.3, G_I the data \mathcal{D} we want to describe using our models \mathcal{M} . As such, informally, our goal is to identify those nodes, and an infection propagation ripple starting from those nodes, by which G_I can most easily be described.

9.3.1 Cost of the Model

As our models we consider *seed sets*. A seed set $\mathcal{S} \subseteq \mathcal{V}_I$ is a subset of $|\mathcal{S}|$ nodes of G_I from which the infection starts spreading—the ‘patients zero’, so to speak. We denote by $\mathcal{L}(\mathcal{S})$ the encoded length, in bits, of a seed set \mathcal{S} .

To describe a seed set \mathcal{S} , we first have to encode how many nodes \mathcal{S} contains. This number, $|\mathcal{S}|$, is upper-bounded by the number of nodes in G . Hence, by using straightforward block-encoding we can encode $|\mathcal{S}|$ in $\log N$ bits, by which we spend equally many bits to encode either a small or a large number. In general, however, we favor small

seeds sets: simple explanations. The MDL optimal Universal code for integers [Ris83] is therefore a better choice as it rewards smaller seed sets by requiring fewer bits to encode their size. With this encoding, $\mathcal{L}_{\mathbb{N}}$, the number of bits to encode an integer $n \geq 1$ is defined as $\mathcal{L}_{\mathbb{N}}(n) = \log^*(n) + \log(c_0)$, where \log^* is defined as $\log^*(n) = \log(n) + \log \log(n) + \dots$, where only the positive terms are included. To make $\mathcal{L}_{\mathbb{N}}$ a valid encoding, c_0 is chosen as $c_0 = \sum_{j \geq 1} 2^{-\mathcal{L}_{\mathbb{N}}(j)} \approx 2.865064$ such that the Kraft inequality is satisfied.

To identify which nodes in G are seed nodes, we use the very efficient class of data-to-model codes [VV04]. A data-to-model code is essentially an index into a canonically ordered enumeration of all possible data (values) given the model (the provided information). Here, we know $|\mathcal{S}|$ unique nodes have to be selected out of N , for which there are $\binom{N}{|\mathcal{S}|}$ possibilities. Assuming a canonical order, $\log \binom{N}{|\mathcal{S}|}$ gives us the length in bits of an index to the correct set of node ids.

Combining the above, we now have $\mathcal{L}(\mathcal{S})$ for the number of bits to identify a seed set $\mathcal{S} \subseteq \mathcal{V}_I$ as

$$\mathcal{L}(\mathcal{S}) = \mathcal{L}_{\mathbb{N}}(|\mathcal{S}|) + \log \binom{N}{|\mathcal{S}|} . \quad (9.1)$$

9.3.2 Cost of the Data given the Model

Next, we need to describe the infected subgraph G_I given a seed set \mathcal{S} . We do this by encoding the infection *propagation ripple*, or the description of ‘what happened’. Starting from the seed nodes, per time step we identify that set of nodes that gets infected at this time step, iterating until we have identified all the infected nodes.¹

Propagation ripples: More formally, a propagation ripple R is a list of node ids per time-step t , which represents the order in which nodes of G_I became infected, starting from \mathcal{S} at time $t = 0$. Let us write $\mathcal{V}_I^t(\mathcal{S}, R)$ to indicate the set of infected nodes at time t starting from seed set \mathcal{S} and following ripple R , with $\mathcal{V}_I^0 = \mathcal{S}$. For readability, we do not write \mathcal{S} and R wherever clear from context. As such, a valid propagation ripple R is a partitioning of node ids $\mathcal{V}_I \setminus \mathcal{S}$ of G_I , where every node in a part is required to have an edge from a node $j \in \mathcal{V}_I^{t-1}$.

Clearly, however, not every ripple from the seed set to the final infected subgraph is equally simple to describe. For instance, the more infected neighbors an uninfected node has, the more likely it is that it will get infected, as it is under constant attack—therefore, it should be more succinct to describe that this node gets infected than it would for a node under single attack.

Frontier sets: To encode a ripple R , at each time t we consider the collection of nodes currently under attack given the SI model (i.e. non-infected nodes with currently atleast one infected neighbor, or if $t = 0$, neighbors to a seed-node $\in \mathcal{S}$). We refer to this set as \mathcal{F}^t , for the frontier-set at time t . Define *attack degree* $a(n)$ of a non-infected

¹When not interested in the actual ripple R , one could encode G_I by its overall probability starting from \mathcal{S} . Obtaining this probability, however, is very expensive, even by MCMC sampling. As we will see in Sections 9.4 and 9.5 computing a good ripple is both cheap and gives good results.

node n as the number of infected neighbor nodes it has at the current iteration, i.e. $a(n) = |\{j \in \mathcal{V} \mid e_{jn} \in \mathcal{E} \wedge X_j(t)\}|$, in which $X_j(t)$ is an indicator function for whether node j is infected at time t .

We divide \mathcal{F}^t into disjoint subsets \mathcal{F}_i^t per attack degree i , that is, into sets of nodes having the same attack degree. As such, we have $\mathcal{F}^t = \mathcal{F}_1^t \cup \mathcal{F}_2^t \cup \dots$, and correspondingly f_1^t, f_2^t, \dots for the sizes of these subsets (we will drop using the t superscript, when clear from context).

Starting from the seed set, for every time step t the receiver can easily construct the corresponding frontier set \mathcal{F}^t —which leaves us to transmit which of the nodes, if any, in the frontier set got infected in the current iteration. As, however, the infection probabilities per attack degree differ, we transmit this information per \mathcal{F}_d^t .

Probability of Infection: The SI model assumes an attack probability parameter β —so, the independent probability p_d of a node in \mathcal{F}_d being infected is: $p_d = 1 - (1 - \beta)^d$. Given p_d we can write down the probability distribution of a total of m_d nodes being infected for each subset \mathcal{F}_d . This is simply a Binomial with parameter p_d i.e.

$$p(m_d \mid f_d, d) = \binom{f_d}{m_d} p_d^{m_d} (1 - p_d)^{f_d - m_d} .$$

Hence, as such, a value for β determines p .

Encoding a Wave of Attack: Given p , a probability distribution for seeing m_d nodes out of f_d infected given an attack degree d , we need $-\log p(m_d \mid f_d, d)$ bits to optimally transmit the value of m_d . That is, we encode m_d using an optimal prefix code—for which we can calculate the optimal code lengths by Shannon entropy [CT06]. Then, once we know both f_d and m_d , we can use code words of resp. $-\log \frac{m_d}{f_d}$ and $-\log 1 - \frac{m_d}{f_d}$ bits long to transmit whether a node in \mathcal{F}_d got infected or not. This gives us

$$\begin{aligned} \mathcal{L}(\mathcal{F}^t) = & - \sum_{\mathcal{F}_d^t \in \mathcal{F}^t} \left(\log p(m_d \mid f_d, d) + m_d \log \frac{m_d}{f_d} \right. \\ & \left. + (f_d - m_d) \log 1 - \frac{m_d}{f_d} \right) \end{aligned} \quad (9.2)$$

for encoding the infectees in the frontier set at time t .

Then, for the recipient to know when to stop reading, we have to transmit how many time steps until we have reached G_I . The number of iterations T will be transmitted just like $|\mathcal{S}|$, using $\mathcal{L}_{\mathbb{N}}$. For the ripple R , starting from the frontier-set defined by the seed nodes \mathcal{S} , we iteratively transmit which nodes got infected at $t + 1$ —which in turn allows the recipient to construct \mathcal{F}^{t+1} . Note that, by $\mathcal{L}(\mathcal{F}^t)$ we assume ripple R to be in time scale of $1/\beta$. That is, for low β we consider a lower time resolution than for high β . This is because the SI model displays a natural invariance of time-scale (see Section 9.2.2). So we have ripple R that gives the infections at every $1/\beta$ time-steps.

With the above, we have $\mathcal{L}(R \mid \mathcal{S})$ for the encoded length of a ripple R starting at a

seed set \mathcal{S} as

$$\mathcal{L}(\mathbf{R} | \mathcal{S}) = \mathcal{L}_{\mathbb{N}}(\mathbf{T}) + \sum_t^T \mathcal{L}(\mathcal{F}^t) \quad . \quad (9.3)$$

9.3.3 The Problem

With $\mathcal{L}(\mathcal{S})$ and $\mathcal{L}(\mathbf{R} | \mathcal{S})$, we have as the total description length $\mathcal{L}(G_I, \mathcal{S}, \mathbf{R})$ of an infected subgraph G_I of G following a valid infection propagation ripple \mathbf{R} starting from a set of seed nodes \mathcal{S} by

$$\mathcal{L}(G_I, \mathcal{S}, \mathbf{R}) = \mathcal{L}(\mathcal{S}) + \mathcal{L}(\mathbf{R} | \mathcal{S}) \quad .$$

Note that as G is constant over all seed sets \mathcal{S} and ripples \mathbf{R} , we can safely ignore it in the computation of the total encoded size, for its encoded length would be constant term and hence not influence the selection of the best model.

By which we can now formally state our problem.

Minimal Infection Description Problem *Given a snapshot of a graph $G(\mathcal{V}, \mathcal{E})$ of N nodes, of which the subgraph $G_I(\mathcal{V}_I, \mathcal{E}_I)$ of N_I nodes are infected, and an infection probability β , by the Minimum Description Length principle we are after that seed set \mathcal{S} and that valid propagation ripple \mathbf{R} for which*

$$\mathcal{L}(G_I, \mathcal{S}, \mathbf{R})$$

is minimal for the Susceptible-Infected propagation strategy.

Clearly, this problem entails a large search space - both in the possible seed-subsets of \mathcal{V}_I and the possible propagation ripples given any seed-set. In fact, as shown by Shah and Zaman [SZ11], even the problem of just finding *one* MLE seed for a given infected snapshot in an arbitrary graph is very hard (#P-Complete, equivalent to counting the number of linear extensions of a poset). Further, the provable algorithms they give are for one seed on d -regular trees only. To tackle the problem on general graphs we hence resort to heuristics.

9.4 Proposed Method

The outline of our approach is as follows: given a fixed number of seeds k , we identify a high-quality k -seed set. Given these seed nodes, we optimize the propagation ripple. With these two combined, we can use our MDL score to identify the best k .

9.4.1 Best seed-set given number of seeds — ‘Exoneration’

A central idea is that intuitively, un-infected nodes should provide some degree of ‘exoneration’ from ‘blame’ for the neighboring infected nodes. See Figure 9.2—it shows two illustrative examples of an infected chain (a) and a chain with a star in the middle (b)

(colored nodes are infected and blue denote the true seeds). Note that while the node X is the most central among the infected nodes and is rightly the most likely seed, the node Y is not a likely seed because of the many un-infected nodes surrounding it. In fact, in this case the most likely starting points would be the two Blue nodes. Hence any method to identify the seed-sets should take into account the centrality of the infected nodes among the infected graph, but also penalize nodes for being too close to the un-infected frontier (the ‘exoneration’). As we explain next, our method is able to do this in a principled manner.

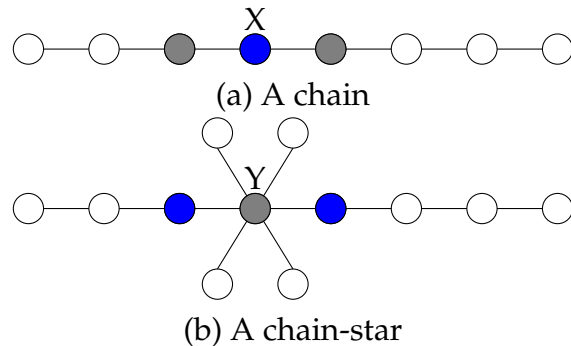


Figure 9.2: Centrality is not enough - effects of ‘exoneration’: Infection snapshot examples (colored nodes are infected, blue nodes are the true seeds) (a) Node X is the most central among the infected nodes; (b) Node Y is the most central among infected nodes, but the high count of non-infected neighbors ‘exonerates’ it.

9.4.2 Finding best single seed—Our Main Idea

We first explain how to find the best single seed and then how to extend it to multiple seeds. Jumping ahead, the main idea is as follows.

Main Idea The single best seed s^* is the one with the highest score in \vec{u}_1 i.e.

$$s^* = \arg \max_s \vec{u}_1(s)$$

where \vec{u}_1 is the *smallest* eigenvector of the laplacian submatrix L_A as defined in Table 9.2. Next, we give the justification.

9.4.3 Finding the best single seed—Justification

From Section 9.3, it is clear that nodes that are not in either the final frontier set \mathcal{F} or \mathcal{V}_I play no role, as they were not infectious nor could have been infected. Hence, WLOG, assume G contains only the infected subgraph G_I and the frontier set \mathcal{F} . Also, assume nodes are numbered in such a way that the first $|\mathcal{V} - \mathcal{V}_I|$ nodes are the un-infected nodes

and the rest are the infected ones. If the total number of nodes in the graph is N , the number of infected nodes is N_I , then the number of un-infected nodes in G is $N - N_I$. Further notation is given in Table 9.2.

Let $X_i(t)$ be the indicator (0/1) Random Variable denoting if node i in the graph is infected or not at time t (1 = infected, 0 = un-infected). Let $Y_{ij}(t)$ be the indicator random variable denoting if node j successfully attacks i at time t . Consider the following update equation for any node $i \in \mathcal{V}_I$:

$$\begin{aligned} X_i(t+1) &= X_i(t) + \\ &(1 - X_i(t)) \times \\ &\quad \bigvee_{j \in \mathcal{N}(i)} Y_{ij}(t)(X_j(t) - X_i(t) + X_i(t)) \end{aligned} \quad (9.4)$$

Following the above equation, if $X_i(t) = 1$ then $X_i(t+1) = 1$, i.e., once a node is infected, it stays infected. Also if $X_i(t) = 0$, then $X_i(t+1) = \bigvee_{j \in \mathcal{N}(i)} Y_{ij}(t)X_j(t)$. Or in other words, an uninfected node may get infected only if an infected neighbor successfully transmits the infection. Additionally for any node $i \in \mathcal{V} - \mathcal{V}_I$, we define $X_i(t) = 0$, as these nodes were not infected at all during the infection process. Hence, the above equations exactly define a discrete-time SI process but with the constraint that the nodes in the given final frontier set *always stay un-infected*, thus enforcing the ‘exoneration’ discussed before. Hence we want to find the seed node which maximizes spread in this ‘constrained’ epidemic, which we show how to next.

For any node $i \in \mathcal{V}_I$, taking expectations both sides of Equation 9.4, and using the fact that for any indicator random variable X , $\mathbb{E}[X] = \Pr(X = 1)$, we get:

$$P_i(t+1) = P_i(t) + U - V \quad (9.5)$$

where,

$$\begin{aligned} P_i(t) &= \Pr(X_i(t) = 1) \\ U &= \mathbb{E} \left[\bigvee_{j \in \mathcal{N}(i)} Y_{ij}(t)(X_j(t) - X_i(t) + X_i(t)) \right] \\ V &= \mathbb{E} \left[X_i(t) \times \bigvee_{j \in \mathcal{N}(i)} Y_{ij}(t)(X_j(t) - X_i(t) + X_i(t)) \right] \end{aligned}$$

Clearly, as all the terms inside are positive,

$$V \geq 0, U \geq 0 \quad (9.6)$$

Also,

$$\begin{aligned} U &\leq \sum_{j \in \mathcal{N}(i)} A(G)_{ij}(P_j(t) - P_i(t) + P_i(t)) \\ &= \sum_{j \in \mathcal{N}(i)} A(G)_{ij}P_i(t) + \sum_{j \in \mathcal{N}(i)} A(G)_{ij}(P_j(t) - P_i(t)) \end{aligned}$$

as an infected node j attacks any of its neighbors i independently with probability $A(G)_{ij}$ (i.e. $\mathbb{E}[Y_{ij}(t)] = A(G)_{ij}$) and because by linearity of expectation, for any two events indicator random variables $\mathbb{1}_A$ and $\mathbb{1}_B$, we have $\mathbb{1}_A \vee \mathbb{1}_B = \mathbb{1}_A + \mathbb{1}_B - \mathbb{1}_A \mathbb{1}_B \Rightarrow \mathbb{E}[\mathbb{1}_A \vee \mathbb{1}_B] \leq \mathbb{E}[\mathbb{1}_A] + \mathbb{E}[\mathbb{1}_B]$. Also note that:

$$\sum_{j \in \mathcal{N}(i)} A(G)_{ij} P_i(t) \leq d_{\max} \times P_i(t) \quad (9.7)$$

where d_{\max} is the largest degree in graph G . Thus,

$$U \leq d_{\max} P_i(t) + \sum_{j \in \mathcal{N}(i)} A(G)_{ij} (P_j(t) - P_i(t)) \quad (9.8)$$

From Equations 9.6 and 9.8, we can conclude that, for each node $i \in \mathcal{V}_I$:

$$\begin{aligned} P_i(t+1) &\leq P_i(t) + d_{\max} P_i(t) \\ &\quad + \sum_{j \in \mathcal{N}(i)} A(G)_{ij} (P_j(t) - P_i(t)) \end{aligned}$$

Let $\sigma = 1 + d_{\max}$. Recall that $\forall t, P_i(t) = 0$ for any *eventual* un-infected node $i \in \mathcal{V} - \mathcal{V}_I$. Let $\vec{P}(t) = [P_1(t), P_2(t), \dots, P_N(t)]^T$ (over all the nodes in \mathcal{V}). Then we can write:

$$\vec{P}(t+1) \leq \sigma \left(I - \frac{1}{\sigma} M \right) \vec{P}(t) \quad (9.9)$$

where, the matrix M (size $N \times N$) is:

$$M = \begin{vmatrix} 0_{N-N_I, N-N_I} & 0_{N-N_I, N_I} \\ 0_{N_I, N-N_I} & L_A \end{vmatrix}$$

where we write $0_{N,M}$ for an all-zeros matrix of size $N \times M$. Let the subvector of $\vec{P}(t+1)$ corresponding to the infected nodes be written as $\vec{P}_I(t+1)$. Then continuing from above and using the upper bound as an approximation, we get:

$$\vec{P}_I(t+1) \approx \sigma \left(I - \frac{1}{\sigma} L_A \right) \vec{P}_I(t) \quad (9.10)$$

$$= \sigma \left(I - \frac{1}{\sigma} L_A \right)^t \vec{P}_I(0) \quad (9.11)$$

$$= \sigma \sum_i \lambda_i^t \vec{u}_i \vec{u}_i^T \vec{P}_I(0) \quad (9.12)$$

where, λ_i and \vec{u}_i are the eigenvalues and eigenvectors of the matrix $I - \frac{1}{\sigma} L_A$. We have the following two lemmas:

Lemma 9.1. *The largest eigenvalue λ_1 and eigenvector \vec{u}_1 of the matrix $I - \frac{1}{\sigma} L_A$ are all positive and real.*

Proof. (Details omitted for brevity) The matrix $I - \frac{1}{\sigma}L_A$ is non-negative, and imagining $I - \frac{1}{\sigma}L_A$ as an adjacency matrix, the corresponding graph is irreducible, as graph G_I (adjacency matrix A) is connected. We then get the lemma due to the Perron-Frobenius theorem [McC00]. \square

Lemma 9.2. *The largest eigenvalue of matrix $I - \frac{1}{\sigma}L_A$ and the smallest eigenvalue of L_A are related as $\lambda_1(I - \frac{1}{\sigma}L_A) = 1 - \frac{1}{\sigma}\lambda_N(L_A)$.*

Proof. (Details omitted for brevity) It is easy to see that any eigenvalue $\text{eig}(I - \frac{1}{\sigma}L_A) = 1 - \text{eig}(\frac{1}{\sigma}L_A)$. As the matrices are symmetric, all the eigenvalues involved are real. By the Cauchy eigenvalue interlacing theorem [Str88] applied to $L(G)$, all the eigenvalues of any co-factor C_{LG} of $L(G)$ are positive. By the famous Kirchoff's matrix theorem [CDS98], the determinant of any co-factor C_{LG} is also non-zero as it counts the number of spanning trees of G . Also, it is well-known that the determinant of any matrix is just the product of its eigenvalues [Str88]. Hence, all eigenvalues of any co-factor matrix C_{LG} of $L(G)$ are strictly positive. We can similarly apply eigenvalue interlacing successively to a suitable C_{LG} and so on till we get to L_A (a principal submatrix of $L(G)$), and get that all eigenvalues of L_A are strictly positive. The lemma follows then. \square

Hence, the eigenvector \vec{u}_1 is also the eigenvector corresponding to the smallest eigenvalue of L_A .

Now, from Equation 9.12 and Lemma 9.1, we have:

$$\vec{P}_I(t+1) = \sigma\lambda_1^t \sum_i \frac{\lambda_i^t}{\lambda_1^t} \vec{u}_i \vec{u}_i^T \vec{P}_I(0) \quad (9.13)$$

$$\approx \sigma\lambda_1^t \vec{u}_1 \vec{u}_1^T \vec{P}_I(0) \quad (9.14)$$

assuming a substantial eigen-gap or 'big-enough' t . Now assuming that $\vec{P}_I(0)$ is all zero *except* for a single seed s for which it is 1, we can conclude that ultimately in our 'constrained' epidemic,

$$\forall i \in \mathcal{V}_I, \Pr(X_i = 1|s) \propto \vec{u}_1(i)\vec{u}_1(s) \quad (9.15)$$

$$\forall i \in \mathcal{V} - \mathcal{V}_I, \Pr(X_i = 1|s) = 0 \quad (9.16)$$

Clearly the most likely single seed s^* would be:

$$s^* = \arg \max_s \left[\sum_{i \in \mathcal{V}_I} \Pr(X_i = 1|s) + \sum_{i \in \mathcal{V} - \mathcal{V}_I} (1 - \Pr(X_i = 1|s)) \right]$$

Using Equations 9.15 and 9.16,

$$\begin{aligned} s^* &\approx \arg \max_s \vec{u}_1(s) \sum_{i \in \mathcal{V}_I} \vec{u}_1(i) \\ &= \arg \max_s \vec{u}_1(s) \end{aligned} \tag{9.17}$$

Hence, for a single seed, we just need to find the node with the largest score in \vec{u}_1 (which is also the smallest eigenvector of the laplacian submatrix L_A from Lemma 9.2).

9.4.4 Finding best k-seed set

Note that simply taking the top-k in the above eigenvector will not give good k-seed-sets due to lack of diversity. This is because the error in the upper-bound approximation used in Equation 9.11 will become larger due to increase in the norm of $\vec{P}_1(0)$. Hence, we treat the newly chosen seed, say s^* , as *un-infected*, effectively exonerating its neighbors and boosting diversity. We redo our computation on the resulting *smaller* infected graph, but a potentially larger frontier set—hence, we take the next best seed *given* the s^* that has already been chosen. So for any given k , we successively find the best next seed, given the previous choices, by removing the previously chosen seeds from the infected set and solving Equation 9.17. For example, in Figure 9.1, the top suspect (Red on the right) will have a lot of suspicious neighbors as well. Thus, using our exoneration technique, the algorithm will be forced away from them towards the remaining Red seed.

9.4.5 Finding a good ripple

As discussed before, once we find the best seed-set \mathcal{S}_k for a given k , we optimize the propagation ripple of \mathcal{S}_k to G_I to minimize the total encoded size. Recall from Section 9.3 that this involves minimizing $\mathcal{L}(R | \mathcal{S})$, which consists of two terms. First, we have the cost of encoding the length of the ripple, the number of time-steps. While \mathcal{L}_N does grow for higher values of T , in practice this term will be dwarfed by the actual encoding of the subsequent frontier sets. As such, minimizing $\mathcal{L}(R | \mathcal{S})$ essentially comes down to minimizing $-\sum_t^T \mathcal{L}(\mathcal{F}^t)$, or, in other words, maximizing the likelihood of the ripple R . Further recall that the SI model has a natural scaling invariance, $1/\beta$. As our score takes this into account, the ripple with the smallest description length should too.

Hence, we design the following procedure. For each attack-degree set \mathcal{F}_d , at any iteration we scale the number of attacks by $1/\beta$ i.e. a set of size f_d is equivalent to a set of size f_d/β . Then, to get the overall MLE ripple, we adopt the following heuristic. We assume that the overall MLE ripple always performs a locally optimal next step. Hence this boils down to choosing the most-likely nodes to get infected at any given step, for a given frontier set \mathcal{F} .

It is well-known that a Binomial distribution $B(n, p)$ has its mode at $\lfloor (n+1)p \rfloor$. Using this fact, at any iteration t , taking into account the scaling, we can see that the most likely

Algorithm 6 NETSLEUTH

Input: $G(\mathcal{V}, \mathcal{E}) \equiv G_I^* \cup \mathcal{F}^*$, $G_I^*(\mathcal{V}_I, \mathcal{E}_I)$ (the infected graph) and \mathcal{F}^* (the frontier set).

Output: \mathcal{S} = the set of seeds (culprits).

- 1: $L(G) = D(G) - A(G)$, the Laplacian matrix corresponding to graph G .
 - 2: $\mathcal{S} = \{\}$
 - 3: $G_I = G_I^*$
 - 4: **while** $\mathcal{L}(G_I, \mathcal{S}, R)$ decreases **do**
 - 5: L_A = the *submatrix* of $L(G)$ corresponding to G_I .
 - 6: v = eigenvector of L_A corresponding to the smallest eigenvalue.
 - 7: $next = \arg \max_i v(i)$
 - 8: $\mathcal{S} = \mathcal{S} \cup \{next\}$
 - 9: R = ripple maximizing likelihood of G_I from \mathcal{S}
 - 10: $G_I = G_I \setminus \{next\}$ (Graph G_I with node $next$ removed)
 - 11: **end while**
 - 12: **return** \mathcal{S}
-

number of nodes infected in an attack-degree set \mathcal{F}_d would be $n_d = \lfloor (f_d/\beta + 1) \times p_d \rfloor$ —where p_d as defined before in Section 9.3 is the attack probability in the set \mathcal{F}_d . As such, we can simply uniformly choose this number of nodes from the \mathcal{F}_d , as each node in \mathcal{F}_d is equally likely to be infected. We do this for every non-empty attack-degree set, for every iteration, until we have infected exactly the observed snapshot. This way, we obtain a most likely propagation ripple for any given seed-set \mathcal{S}_k and can subsequently score it using MDL.

Finally, we stop getting more seeds when the MDL score for \mathcal{S}_k increases as we increase k . Algorithm 6 gives the pseudo-code and Lemma 9.3 shows the running time for our algorithm NETSLEUTH.

Lemma 9.3 (Running Time of NETSLEUTH). *The time complexity of NETSLEUTH is $O(k^*(\mathcal{E}_I + \mathcal{E}_F + \mathcal{V}_I))$.*

Proof. We keep finding \mathcal{S}_k for each seed-set size until MDL tells us to stop. Hence the running time is $O(k^*(\mathcal{E}_I + T_{\text{RIPPLE}} + T_{\text{MDL}}))$, if k^* is the optimal seed-set size and T_{MDL} is the running time of computing the MDL score given the seed set size is k^* . Here we used the fact that calculating the eigenvector using the Lanczos method is approximately $O(E)$ (# edges) for sparse graphs.

The worst-case complexity T_{MDL} of calculating $\mathcal{L}(G_I, \mathcal{S}, R)$ for a given G_I , \mathcal{S} , and R , is $O(\mathcal{E}_I + \mathcal{E}_F + \mathcal{V}_I)$. The $\mathcal{L}(\mathcal{S})$ term is $O(1)$. For the $\mathcal{L}(R | \mathcal{S})$ term, we need to iterate over the ripple, which is at most \mathcal{V}_I steps long. We only have to update the frontier set \mathcal{F} when one or more nodes got infected, for which we then have to update the attack degrees of the nodes connected to the nodes infected at time t . Hence we traverse every edge in $\mathcal{E}_I + \mathcal{E}_F$, and every node in \mathcal{V}_I , which gives it the complexity of $O(\mathcal{E}_I + \mathcal{E}_F + \mathcal{V}_I)$.

Finally, the running time T_{RIPPLE} of computation of the MLE ripple for a given S_k is also $O(\mathcal{E}_I + \mathcal{E}_F + \mathcal{V}_I)$.

So the overall complexity of NETSLEUTH is $O(k^*(\mathcal{E}_I + \mathcal{E}_F + \mathcal{V}_I))$. \square

Hence NETSLEUTH is *linear* in the number of edges and vertices of the infected sub-graph and the frontier set, which makes our method scalable for large graphs (as compared to the methods in [SZ11, LTGM10] which, even for detecting a *single seed*, are $O(N^2)$).

9.5 Experiments

Here we experimentally evaluate NETSLEUTH, in particular its effectiveness in finding culprits—whether it correctly identifies (a) *how many* as well as (b) *which ones*—and its (c) scalability.

9.5.1 Experimental Setup

We implemented NETSLEUTH in Matlab, and in addition we implemented the SI model as a discrete event simulation in C++. All reported results are averaged over 10 independent runs (so we generate 10 graphs for each seed set).

In our study we use both synthetic and real networks—we chose synthetic networks exemplifying different types of situations. We consider the following networks:

1. GRID is a 60×60 2D grid as shown in Figure 9.1.
2. CHAIN-STAR It is a graph of total 107 nodes. The first 7 nodes form a linear chain and the middle node has 100 additional neighbors.
3. AS-OREGON The Oregon AS router graph which is a network graph collected from the Oregon router views. It contains 15 420 links among 3 995 AS peers.²

For the experiments on AS-OREGON, we ran the experiments for true-seed count $k^* = 1, 2, 3$. So for each seed-set, we run a simulation till at least 30% of the graph is infected, and give the resulting footprint as input to NETSLEUTH. Note that, the larger the number of infections, the tougher it is to find the true seeds, as in the SI model any seed will eventually infect the whole graph with certainty. Finally, we make sure that the infected sub-graph was connected—otherwise, we just have separate problem instances.

As discussed in the introduction and Section 9.6, the existing proposals for identifying culprits consider significantly different problems settings than we do (see Table 9.1); the Rumor Centrality of Shah and Zaman [SZ10, SZ11] can only discover *one* seed node, while Effectors of Lappas et al. [LTGM10] even consider a completely different infection model. As such we can not meaningfully compare performances, and hence here only consider NETSLEUTH.

²For more information see <http://topology.eecs.umich.edu/data.html>.

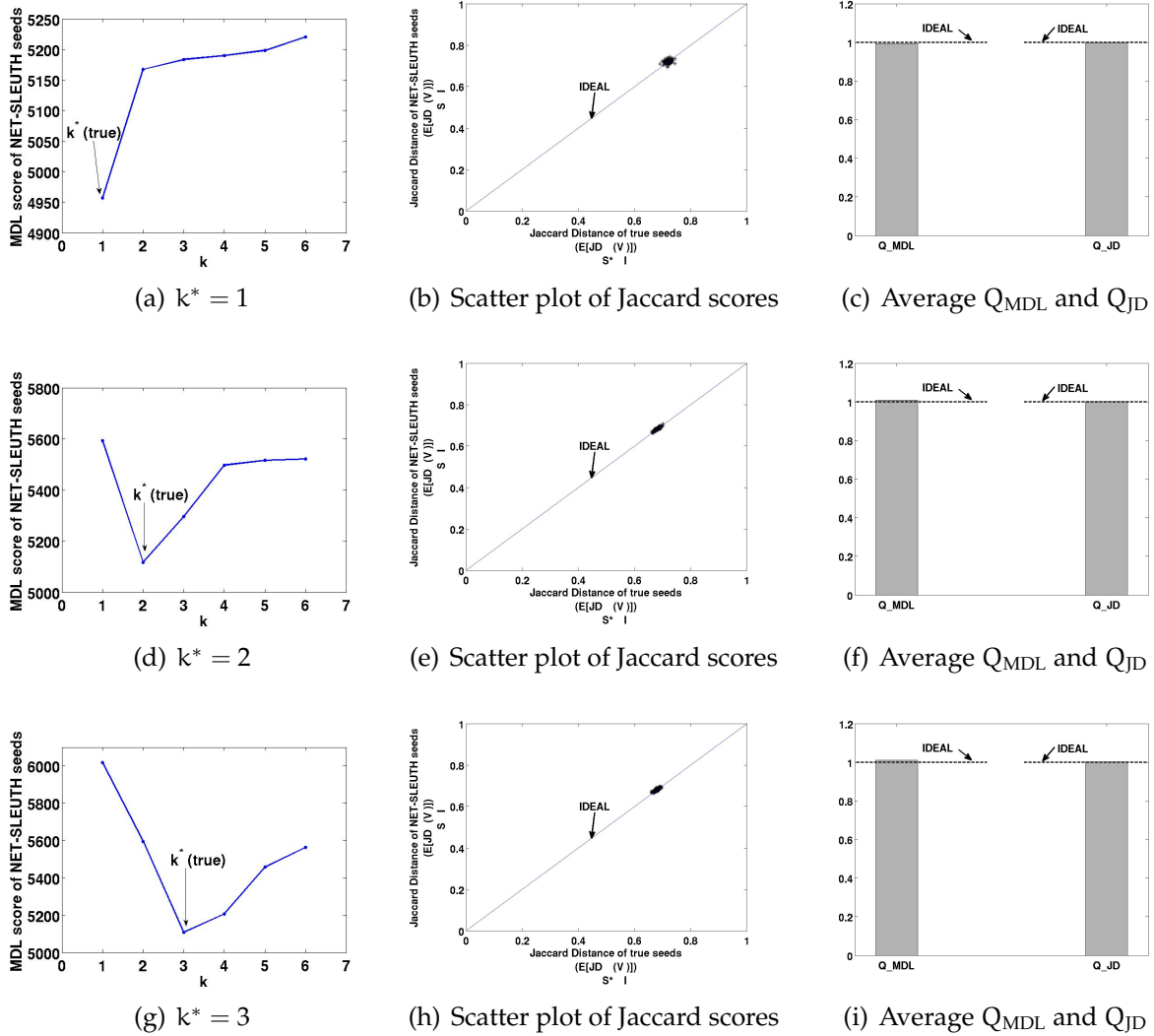


Figure 9.3: Effectiveness of NETSLEUTH in answering both *How many* and *Which ones* - Results of our experiments on the AS-OREGON graph for true-seed-count $k^* = 1, 2, 3$ (rows, subfigures (a-c), (d-f), (g-i) respectively). First column, (a),(d),(g), MDL scores as a function of k found by NETSLEUTH are near-convex; also we recover the true number in all cases. Second column, (b),(e),(h), scatter plots of Jaccard scores ($JD_x(V_I)$) of NETSLEUTH seeds (y-axis) and the corresponding true seeds (x-axis). On or below the 45-degree line is better. Each point average of 10 runs. Note that for many runs the seeds identified by NETSLEUTH score exactly or even better than the true seeds. Third column, (c),(f),(i), average Q_{MDL} and Q_{JD} scores for the seeds returned by NETSLEUTH. Each bar represents the average over 90 different seed-sets. Note that all the bars are close to 1, indicating that we consistently find high-quality seed sets both with the Jaccard measure, and with the MDL measure.

Evaluation Function—a subtle issue

How to evaluate the goodness of a seed set? That is, in Figure 9.1, how close are the red seeds (recovered) from the blue seeds (true)? Notice that the recovered seeds may actually have *better* score than the actual ones, for the same reason that the sample mean of a group of 1D Gaussian instances gives lower sum-squared-distances than the theoretical mean of the distribution. Moreover, even for evaluation, it is intractable to compute the exact probability of observing the footprint from a given seed-set.

Thus we propose two quality measures. The first, Q_{MDL} , is based on our MDL: we report the ratio of the MDL score of our seeds, vs. the MDL score of the actual seeds i.e.

$$Q_{\text{MDL}} = \frac{\mathcal{L}(G_I, \mathcal{S}, R)}{\mathcal{L}(G_I, \mathcal{S}^*, R^*)} \quad (9.18)$$

Clearly, the closer to 1, the better.

The second Q_{JD} intuitively measures the *overlap* of the footprint produced by a seed-set \mathcal{S} and the input footprint $G_I(\mathcal{V}_I, \mathcal{E}_I)$. Clearly, the candidate seed-set \mathcal{S} can produce n footprints, when we run n simulations; so we compute $\mathbb{E}[JD_{\mathcal{S}}(\mathcal{V}_I)]$, the average Jaccard distance³ of all these n footprints, w.r.t. the true input footprint \mathcal{V}_I . As with Q_{MDL} , we normalize it with the corresponding score $\mathbb{E}[JD_{\mathcal{S}^*}(\mathcal{V}_I)]$ for the true seed-set, and thus report the ratio,

$$Q_{\text{JD}} = \frac{\mathbb{E}[JD_{\mathcal{S}}(\mathcal{V}_I)]}{\mathbb{E}[JD_{\mathcal{S}^*}(\mathcal{V}_I)]} \quad (9.19)$$

Again, the closer to 1, the better.

9.5.2 Effectiveness of NETSLEUTH in identifying How Many

In short, NETSLEUTH was able to find the exact number of seeds for all the cases. Figures 9.3(a),(d),(g) show the MDL score as a function of $k = 1, 2, \dots, 6$ seeds found by NETSLEUTH before stopping, for true seed-sets with (a) $k^* = 1$, (b) $k^* = 2$ and (c) $k^* = 3$ respectively on the AS-OREGON network. Note that the plots show near-convexity, with the minimum at the true k^* , justifying our choice of stopping after $j = 6$ iterations of increasing scores. It also shows the power of our approach, as we can easily recover the true number of seeds using a principled approach.

9.5.3 Effectiveness of NETSLEUTH in identifying Which Ones

In short, in addition to finding the correct number of seeds, NETSLEUTH is able to identify good-quality seeds with high accuracy. For our synthetic graphs, NETSLEUTH is able to point out that there must have been exactly 2 seeds for both the GRID and CHAIN-STAR

³We use the standard definition of Jaccard Distance between two sets \mathcal{A} and $\mathcal{B} = 1 - \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|}$.

examples—respectively identified as the Red circles in Figure 9.1, and the Blue nodes in Figure 9.2(b)), agreeing with the ground-truth and intuition.

Figures 9.3(b-c),(e-f),(h-i) show the results of our experiments for different number $k^* = 1, 2, 3$ of true seeds on the AS-OREGON graph. We randomly selected 90 seed-sets of each size. We made sure that the seed-sets contained both well-connected and weakly connected nodes. Each point is an average of 10 runs.

Firstly, although not shown in the figures, NETSLEUTH was able to perfectly recover the true number of seeds in almost all cases. For each seed-set, we calculate $JD_s(\mathcal{V}_1)$ for the seeds returned by NETSLEUTH and the true seeds and give the scatter plot in Figures 9.3(b)(e)(h) for true-seed count $k^* = 1, 2, 3$ respectively (rows). Hence points on or below the 45-degree line (solid blue) are better. Clearly, almost all points are concentrated near the diagonal, showing high quality. In fact, many points are exactly on the line, meaning we are able to recover the true seeds *perfectly* for many cases. We do not show similar plots with our MDL score due to lack of space.

Next, we calculate Q_{MDL} and Q_{JD} averaged over all the different seed-sets (of the same size for $k^* = 1, 2, 3$). Results are shown in the bar plots (third column) of Figures 9.3(c)(f)(i). The true-seed scores are represented by the dotted line at 1, for both Q_{MDL} and Q_{JD} . Clearly, all of bars are close to 1, demonstrating that NETSLEUTH consistently finds very good culprits. Moreover, both Q_{MDL} and Q_{JD} quality metrics are similar in magnitude for all k^* 's — increasing our confidence in our results.

9.5.4 Scalability

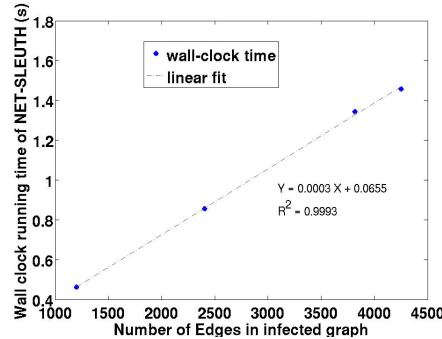


Figure 9.4: NETSLEUTH Scalability: Wall-clock running time (in seconds) for increasingly larger infected snapshots of AS-OREGON (as the complexity just depends on the size of the snapshot) $k^* = 1$. Each point average of 10 runs. Note that, as expected, the running time scales linearly.

Figure 9.4 demonstrates the scalability of NETSLEUTH after running it on increasingly larger infected snapshots of AS-OREGON (as the complexity just depends on the size of the snapshot). As expected from our Lemma 9.3, the running-time is linear on the number of edges of the infected graph.

9.6 Related Work

As mentioned in the introduction, although diffusion processes have been widely studied, the problem of ‘reverse engineering’ the epidemic has not received much attention, except papers by Shah and Zaman [SZ10, SZ11] and Lappas et al. [LTGM10]. Shah and Zaman [SZ10, SZ11] formalized the notion of *rumor-centrality* for identifying the single source node of an epidemic under the SI model, and showed an optimal algorithm for *d-regular trees*. Lappas et al. [LTGM10] study the problem of identifying k seed nodes, or *effectors* of a partially activated network, which is assumed to be in *steady-state* under the IC (Independent-Cascade) model. In contrast, we allow for (a) multiple seed nodes, (b) a snapshot from any time during the infection, and (c) find the number of seeds automatically, even for general graphs. Finally we are also more efficient with linear time on edges of the infected graph.

Rest of the related work into areas deals with epidemic/cascade-style processes and problems related to them like epidemic thresholds, immunization and influence maximization (most of which we have already seen previously).

Influence Maximization An important problem under the viral marketing setting is the influence maximization problem [RD02, KKT03, GLL11, CWW10, HBW11]. Another remotely related work is outbreak detection [LKG⁺07] in the sense that we aim to select a subset of ‘*important*’ nodes on graphs.

MDL We are not the first to use the Minimum Description Length principle [Gr7] for a data mining purpose. Faloutsos and Megalooikoumou [FM07] argue many data mining problems are related to Kolmogorov Complexity, which means they can be practically solved through compression—examples include clustering [CV05], pattern mining [VvS11], and community detection [CPMF04]. We are, to the best of our knowledge, however, the first to employ MDL with the goal of identifying culprits.

9.7 Conclusions

In this chapter we discussed finding culprits, the challenging problem of identifying the nodes from which an infection in a graph started to spread. We proposed to employ the Minimum Description Length principle for identifying that set of seed nodes from which the given snapshot can be described most succinctly. We introduced NETSLEUTH (based on a novel ‘submatrix-laplacian’ method), a highly efficient algorithm for both identifying the set of seed nodes that best describes the given situation, and *automatically* selecting the best number of seed nodes—in contrast to the state of the art.

Experiments showed NETSLEUTH attains high accuracy both in detecting the seed nodes, and correctly identifying their number. Importantly, NETSLEUTH scales *linearly* with the number of edges of the infected graph, $O(\mathcal{E}_F + \mathcal{E}_I + \mathcal{V}_I)$, making it applicable on large graphs.

Part III

Models

Overview

In this part, we turn our attention to building expressive models from real-data for various propagation scenarios, using insights gained so-far. Apart from having explanatory power, we will see how these models can also perform challenging tasks like forecasting and anomaly detection. We address two main settings here:

- **Models for Information Diffusion:** How quickly does a piece of news spread over these media? Do the rise and fall activity patterns in online media have any underlying ‘laws’? After analyzing numerous real datasets, we present SPIKEM, a general, accurate and succinct model that explains such rise-and-fall patterns, unifies previous models and patterns, is interpretable enabling effective forecasting of trends, and answering ‘what-if’ scenarios.
- **Models for Competing Tasks:** If ‘Alice’ has $n_A=50$ contacts and did $n_B=100$ phone-calls to them, what should we expect for ‘Bob’, who has twice the contacts? One would expect a linear relationship (double the contacts, double the phone-calls). However, we show that in numerous settings (like tweets vs re-tweets), the relationship is a power law, being sub- or super-linear. We also present a simple model, CTM, based on competing species which succinctly explains the prevalence of such power-laws, and then subsequently use to spot outliers like telemarketers as well.

Chapter 10

Rise and Fall in Information Diffusion

The recent explosion in the adoption of search engines and new media such as blogs and Twitter have facilitated faster propagation of news and rumors. How quickly does a piece of news spread over these media? How does its popularity diminish over time? Does the rising and falling pattern follow a simple universal law?

In this chapter, we propose SPIKEM, a concise yet flexible analytical model for the rise and fall patterns of influence propagation. Our model has the following advantages: (a) unification power: it generalizes and explains earlier theoretical models and empirical observations; (b) practicality: it matches the observed behavior of diverse sets of real data; (c) parsimony: it requires only a handful of parameters; and (d) usefulness: it enables further analytics tasks such as forecasting, spotting anomalies, and interpretation by reverse-engineering the system parameters of interest (e.g. quality of news, count of interested bloggers, etc.).

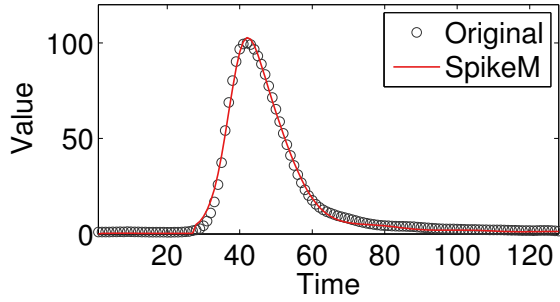
Using SPIKEM, we analyzed 7.2GB of real data, most of which were collected from the public domain. We have shown that our SPIKEM model accurately and succinctly describes all the patterns of the rise-and-fall spikes in these real datasets.

10.1 Introduction

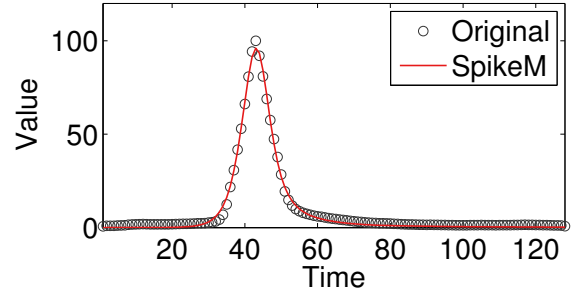
How do spikes behave in social media? Online social media is spreading news and rumors in new ways and search engines have facilitated such spreading magnificently, creating bursts and spikes. Some rumors (or memes, hashtags) start slowly and linger; others spike early and then decay; others show more complicated behavior, as we show in Figure 10.1.

Do real rise-and-fall patterns have any qualitative differences? Do they form different classes? If yes, how many? Earlier work on Youtube data claims there are four classes [CS08]. Empirical work found six classes [YL11]. How many classes are there after all?

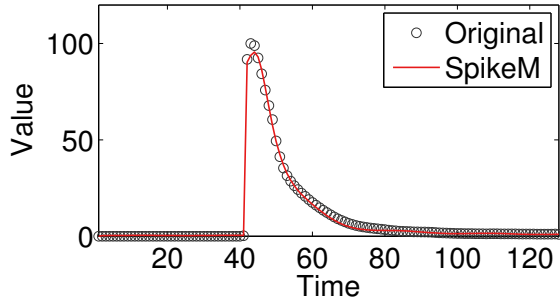
Our answer is: *one*. We provide a unifying model, SPIKEM, that requires only a handful of parameters, and we show that it can generate all patterns found in real data



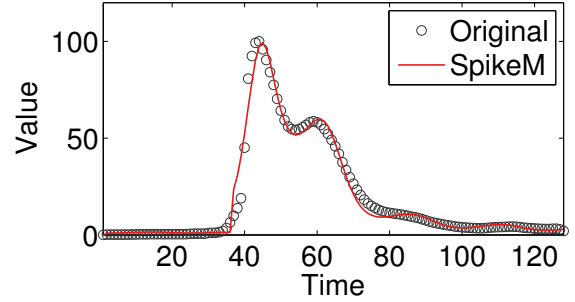
(a) Pattern C1



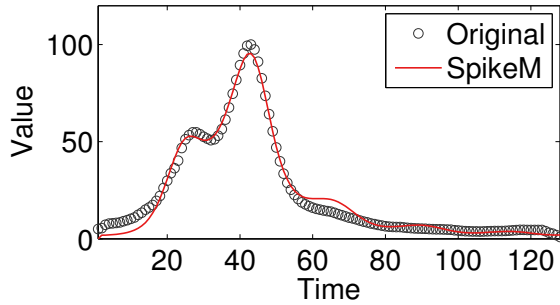
(b) Pattern C2



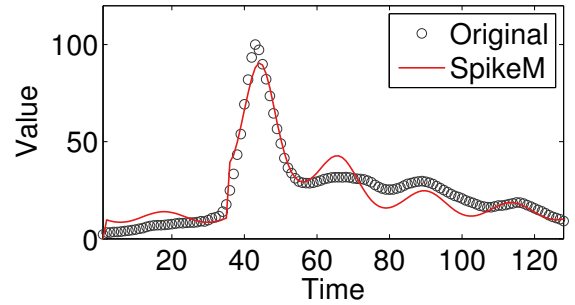
(c) Pattern C3



(d) Pattern C4



(e) Pattern C5



(f) Pattern C6

Figure 10.1: Modeling power of SPIKEM: six types of spikes (K-SC from [YL11]) shown as dots, and our model fit in solid red line. Data sequences span over 120 time-ticks, while SPIKEM requires only seven parameters. The fit is so good, that the red line is often invisible, due to occlusion.

simply by changing the parameter values.

Figure 10.1 shows six representative spikes of online media (memes) from K-SC [YL11], as gray circles, as well as our fitted model, as a solid red line. Notice that the fitting is very good, despite the fact that our SPIKEM model requires only seven parameters, and that the time-sequences span 120 intervals.

Informally, the problem we want to solve is to model/predict an activity (e.g., number of blog postings), as a function of time, given some breaking-news at a given timetick. We will use a blogger example for brevity and clarity, but many other processes could be also modeled (people buying products, computer viruses infecting machines, rumors spreading over Twitter, etc). Thus, we have:

Informal Problem 1 (what-if). **Given** a network of bloggers (/hosts/buyers), a shock (e.g., event) at time n_b , the interest/quality of the event, the count S_b of bloggers that immediately (= time n_b) blog about the event, **find** how the blogging activity will evolve over time.

A closely related problem is to develop a parsimonious model, that can be made to fit several spikes observed in the past (as we do in Figure 10.1). That is,

Informal Problem 2 (model design). **Given** the behavior of several spikes in the past, **find** an equation/model that can explain them, with as few parameters as possible.

It would be good if the parameters had an intuitive explanation (like, ‘number of bloggers’, ‘quality of news’, etc, as opposed to, say, α_1 , α_2 of an autoregressive model (AR/ARIMA)).

In this work, we propose SPIKEM model to solve both of the aforementioned problems. Our SPIKEM has the following advantages:

- **Unification power:** it includes earlier patterns and models as special cases ([YL11, LBK09]),
- **Practicality:** it matches the behavior of numerous, diverse, real datasets, including power-law decay
- **Parsimony:** our model requires only a handful of parameters
- **Usefulness:** thanks to the SPIKEM model, we can answer ‘what-if’ questions (see subsection 10.5.1), spot outliers, reverse-engineer the system parameters (quality of news, count of interested bloggers, time-of-day behavior of bloggers)

Our SPIKEM model is enabled by a careful design to incorporate (a) the power-law decay in infectivity, (b) a finite population, and (c) proper periodicities. Earlier models ignored one or more of the above issues.

Thanks to the *practicality* of SPIKEM, we can make forecasting, analysis of ‘what-if’ scenarios, and detection of anomalies, as we show in section 10.4 and section 10.5. We should highlight that traditional AR, ARIMA and related linear models are fundamentally unsuitable, because they are *linear* (and can diverge to infinity) and because they lead to exponential decays (as opposed to the power law that reality seems to obey). Table 10.1 illustrates the relative advantages of our method: the C-S method (Crane and

	C-S	K-SC	SI	AR	SPIKEM
System identification			✓		✓
Non-linear			✓		✓
Power law decay	✓				✓
Periodicity				✓	✓
Forecasting				✓	✓

Table 10.1: Capabilities of approaches. Only our approach meets all specs.

Sornette) [CS08] assumes an infinite population of bloggers; the clusters in K-SC [YL11] (repeated in Figure 10.1) are non-parametric and are incapable of forecasting. The SI model (closely related to the Bass model [Bas69] of the market penetration of new products) leads to exponential decay, as opposed to the power-law decay that we observe in real data.

Outline The rest of the chapter goes as follows: Section 10.2 presents an overview of the related work and Section 10.3 the proposed model. Sections 10.4 and 10.5 show our experimental results on a variety of datasets. We conclude in section 10.7.

10.2 Background

In this section, we present the fundamental concepts.

Epidemiology fundamentals. The most basic epidemic model is the so-called ‘Susceptible-Infected’ (SI) model. Each object/node is in one of two states - Susceptible (S) or Infected (I). Each infected node attempts to infect each of its neighbors independently with probability β , which reflects the strength of the virus. Once infected, each node stays infected forever. If we assume that the underlying network is a clique of N nodes, and use our notation (‘B’ for blogged = infected) the most basic form of the model is:

$$\frac{dB(t)}{dt} = \beta * (N - B(t))B(t) \quad (10.1)$$

where the time t is considered continuous, dB/dt is the derivative, and the initial condition reflects the external shock (say, $B(0) = b$ externally infected people). The justification is as follows: β is the strength of the virus, that is, the probability that an encounter between an infected person (‘B’) and an uninfected one, will end up in an infection - and we have $B * (N - B)$ such encounters. The solution for $B()$ is the sigmoid, and its derivative is symmetric around the peak, with an exponential rise and an exponential fall (we discuss later in Figure 10.2). There we also show the weakness of the SI model: real data have a power-law ‘fall’ pattern.

Self-excited Hawkes process. Crane et al. [CS08] used a self-excited Hawkes conditional Poisson process [HO74] to model YouTube views per day, showing that spikes in the activity have a power-law rise pattern, and a power-law fall pattern, depending

on the model parameters. Roughly, the Hawkes process is a Poisson process where the instantaneous rate is not constant, but depends on the count of previous events, whose effect drops with the age τ of the event. That is, if there were a lot of events (viewings/bloggings) recently, we will have many such events today.

The base model states that the rate of spread of infection depends on (a) the external source $S(t)$ and (b) self-excitation, that is, on earlier-infected nodes ($i = 1, \dots$); these nodes spread the infection with decaying virus strength $\phi(\tau)$, their age τ grows, times some constant μ_i . The constant μ_i is equivalent to the degree of the infected node i .

$$\frac{dB(t)}{dt} = S(t) + \sum_{i, t_i \leq t} \mu_i \phi(t - t_i) \quad (10.2)$$

The model typically assumes that the μ_i values are equal, namely that all nodes have the same degree ('homogeneous' graph). It also silently assumes that there are infinite nodes available for infection, and it may actually diverge to infinity.

Next we present our SPIKEM model, which avoids the shortcomings of the SI and Hawkes models, and has several more desirable properties.

10.3 Proposed Method

In this section we present our proposed method, analyze it, and we provide the reader with several interesting -at least in our opinion- observations.

Our model tries to capture the following behaviors, that we observed with several of our real data

- P1: power-law fall pattern
- P2: periodicities

and at the same time we want to

- P3: avoid the divergence to infinity

that other models may have. To handle P3 (divergence), we force our model to have a finite population, and adjust the equations accordingly. To handle P1 (power-law fall pattern), we assume that the infectivity of a node (= popularity of a blog post) decays with the INFLUENCE EXPONENT, which we set at -1.5. The handling of periodicities is discussed in subsection 10.3.2.

We describe our model in steps, adding complexity, and we start with the base model.

Preliminaries We assume there are N bloggers, and none of them is yet blogging about the topic of interest. At time n_b , an event happens (such as the 2004 Indonesian tsunami, or a controversial political speech such as 'lipstick on a pig'), and S_b bloggers immediately blog about it. We refer to this external event as a *shock*, and n_b and S_b are the birth-time and the initial magnitude of the shock.

Our model needs a few more parameters: the first is the quality/interestingness of the news, which we refer to as β , since this is the standard symbol for the infectivity of a virus in epidemiology literature. If β is zero, nobody cares about this specific piece of news; the higher the value, the more bloggers will blog about it.

Finally, we have the decay function $f(n)$, which models how infective/influential a blog posting is, at age n . Standard epidemiology models assume that $f()$ is constant (once sick, you have the same probability of infecting others); recent analysis has shown that the influence drops with age, following a power law.

The above are the parameters of the base model. Before we list the equations, we want to briefly mention a derived quantity, $\beta * N$; this quantity roughly corresponds to the R_0 ('R-naught') found in the epidemiology literature. This tells us the size of the "first burst": if only one person was infected, how many would be infected in the next time-tick?¹

In summary, the scenario we model is as follows:

- nothing happens, until a news-event appears, at birth-time n_b .
- S_b bloggers immediately blog about it.
- other bloggers visit the initial S_b (or follow-up) bloggers, and occasionally get 'infected' and blog about the event, too.

We also assume that

- each blogger blogs at most once about the event
- no other related event occurs - that is, the shock function $S()$ has only one spike.

Without loss of generality, we also assume that once an un-informed blogger sees an infected/informed blog, he/she always blogs about the event (if he/she blogs with probability $\rho < 1$, we could absorb ρ in the infectivity factor β)

Our goal is to find an equation to describe the number $\Delta B(n)$ of people blogging at time-tick n , as a function of n and of course the system parameters (total number of bloggers N , strength of infection β etc).

10.3.1 Base model - SPIKEM-BASE

The model we propose has nodes (=bloggers) of two states:

- U: Un-informed of the rumor
- B: informed, and Blogged about it

For those who just got informed at time-tick n , we'll use the symbol $\Delta B(n)$, and we assume that, once informed, a person will blog about the rumor immediately.

Let $U(n)$ be the number of un-informed people at time n , and let $\Delta B(n)$ the number of people that just found out about the rumor at time n , and blogged immediately about it.

¹yes, it should be $N - 1$, but we sacrifice accuracy, for intuition.

Model 1 (SPIKEM-Base). *Our base model is governed by the equations*

$$\Delta B(n+1) = U(n) \cdot \sum_{t=n_b}^n (\Delta B(t) + S(t)) \cdot f(n+1-t) + \epsilon \quad (10.3)$$

$$U(n+1) = U(n) - \Delta B(n+1) \quad (10.4)$$

where

$$f(\tau) = \beta * \tau^{-1.5} \quad (10.5)$$

and initial conditions:

$$\Delta B(0) = 0, \quad U(0) = N$$

In addition, we add an external shock $S(n)$, a spike generated at birth-time n_b . Mathematically, it is defined as follows:

$$S(n) = \begin{cases} 0 & (n \neq n_b) \\ S_b & (n = n_b) \end{cases} \quad (10.6)$$

Justification of the model We do it in steps:

- The term $\Delta B(t) + S(t)$ captures the count of bloggers plus external sources, that got activated at time-tick t ; their infectivity is modulated by the $f()$ infectivity function, since we assume that the infectivity of a source/blogger decays with time. The summation is over all past time-ticks since the birth-time n_b of the shock.
- The infectivity function $f()$ exactly follows a power law with exponent -1.5 as discovered by earlier work on read data: real bloggers [LMF⁺07], and response to mails by Einstein and Darwin [Bar05].
- The meaning of the summation is the available stimuli at time-tick n ; the available targets are the un-informed bloggers $U(n)$, and the product gives the number of new infections.
- We add a noise term ϵ to handle cases such as hashtag 'egypt' on Twitter: some people tweet about Egypt anyway, but a large shock occurred during the events in Tahrir square. Very often, $\epsilon \simeq 0$.

This completes the justification of our base model.

We also mention some facts that our model obeys: by definition

$$B(n) = \sum_{t=0}^n \Delta B(t)$$

and of course we have the invariant

$$B(n) + U(n) = N$$

where N is the total number of people/bloggers.

Symbol	Definition
N	total population of available bloggers
n_d	duration of sequence
n	time-tick ($n = 0, \dots, n_d$)
$U(n)$	count of un -informed bloggers
$B(n)$	count of informed b loggers
$\Delta B(n)$	delta: count of informed b loggers at exactly time n
$f(n)$	inf _e ctiveness of a blog-post, at age n
β	strength of infection
$\beta * N$	“first-burst” size of infection
$S(n)$	volume of external s hock at time n
n_b	starting time of b reaking news
S_b	strength of external shock at birth (time n_b)
ϵ	background noise
P_a	strength of periodicity
P_p	period
P_s	phase shift of periodicity

Table 10.2: Symbols and definitions

10.3.2 With periodicity - SPIKEM

Bloggers may modulate their activity following a daily cycle (or weekly, or yearly). For example, among the $U(n)$ uninformed bloggers at time n , a fraction of them are not paying attention (say, because they are tired or asleep). How can we reflect this in our equations? We propose an answer below, and then we provide the justification.

Model 2 (SPIKEM). *We can capture the periodic behavior of bloggers with the following equations:*

$$\Delta B(n+1) = p(n+1) \cdot \left(U(n) \cdot \sum_{t=n_b}^n (\Delta B(t) + S(t)) \cdot f(n+1-t) + \epsilon \right) \quad (10.7)$$

$$p(n) = 1 - \frac{1}{2} P_a \left(\sin\left(\frac{2\pi}{P_p}(n + P_s)\right) + 1 \right) \quad (10.8)$$

where $U(n)$, $S(t)$ and $f(n)$ are defined in Model 1.

Justification The model is identical to SPIKEM-base, with the addition of the periodicity factor $p(\cdot)$. This captures the fact that bloggers tone down their activity, say, during the night, or even stop it altogether. The idea is that $U(\cdot)$ is the count of victims available

for infection, and the summation is the number of attacks. Under normal circumstances, each victim-attack pair would lead to a new victim; however, since the victims are not paying full attention (tired/asleep), the attacks are not so successful, and thus we prorate them by the $p()$ periodic function.

- P_p stands for the period of the cycle (say, 24 hours).
- P_s stands for the phase shift: if the peak activity is at noon, and the period is $P_p=24$ hours, then $P_s=18$.
- P_a depends on the amplitude of the fluctuation, and specifically it gives the relative value of the off-time (say, midnight), versus peak time (say, noon). Thus, if $P_a=0$, we have no fluctuation.

10.3.3 Additional details

Model extensions We could easily extend our model so that it has several shocks as opposed to just one as considered here. We could also extend it to have multiple cycles (daily, weekly, yearly). We do not elaborate on these extensions for two reasons: (a) for clarity and (b) because the current model fits real data very well, anyway.

Learning the parameters Our model consists of a set of seven parameters:

$$\theta = \{N, \beta, n_b, S_b, \epsilon, P_a, P_s\}.$$

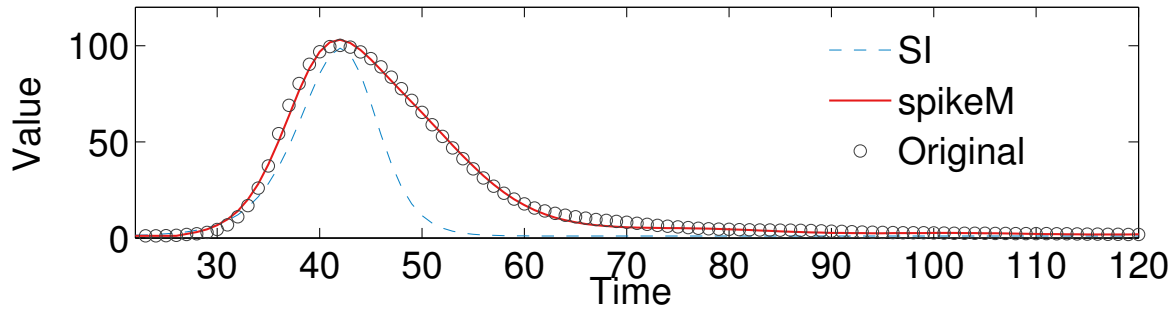
Given a real time sequence $X(n)$ of bloggers at time-tick n ($n = 1, \dots, n_d$), we use *Levenberg-Marquardt (LM)* [Lev44] to minimize the sum of the errors:

$$D(X, \theta) = \sum_{n=1}^{n_d} (X(n) - \Delta B(n))^2.$$

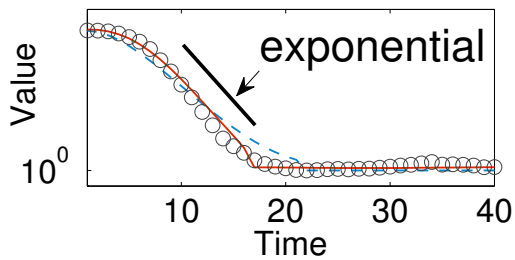
Analysis - exponential rise, power-law fall It is not obvious from the equations of our model, but its rise pattern is exponential, while the fall pattern obeys a power law. This is desirable, because this behavior seem to be prevailing in real data, as we show in Figure 10.2. Let n_{mode} denote the time-tick at which the wave $\Delta B()$ reached its maximum volume. By *rise plot* we mean the plot of values from the birth-time n_b until n_{mode} (and reversing time $abs(n - n_{mode})$) The *fall-plot* is defined similarly: activity $\Delta B()$ versus delay from the peak $n - n_{mode}$. Notice that there is a power law for the fall, and an exponential shape for the rise. We also show the traditional ‘SI’ model, which, as expected, exhibits exponential behavior for both rise and fall.

10.4 Experiments

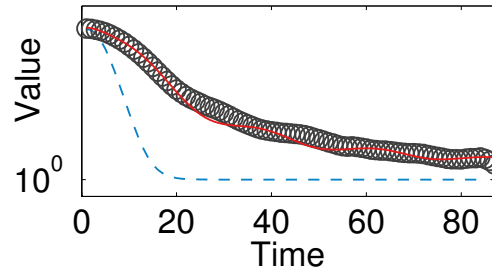
To evaluate the effectiveness of SPIKEM, we carried out experiments on real datasets. The experiments were designed to answer the following questions:



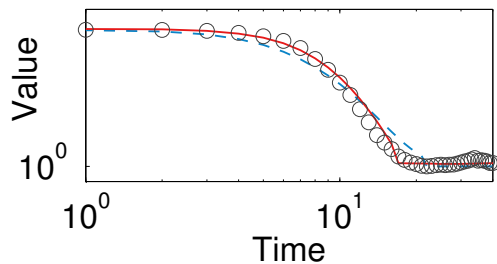
(a) Whole sequence (linear-**log** scale) duration=120, peak at $n_{mode} = 42$



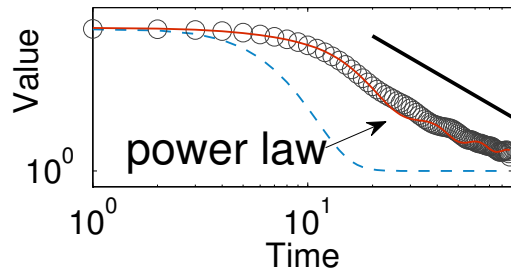
(b) Rise-plot (linear-**log** scale)
Time n:42, 41, ... 1



(c) Fall-plot (linear-**log**)
Time n:42, 43, ...120



(d) Rise-plot (**log-log** scale)
Time n:42, 41, ... 1



(e) Fall-plot (**log-log**)
Time n:42, 43, ...120

Figure 10.2: Fitting results of SPIKEM vs. SI for pattern C1 in Figure 10.1. The original sequence (in gray circles), and our model (red line) have an exponential rise and a power-law drop; the SI model (blue dashed line) is exponential on both and thus unrealistic. Top row: full interval; left column: only the rise part; right column: only the 'fall' part.

- Q1: Can we explain the cluster centers of K-SC?
- Q2: How well do we match *MemeTracker* data?
- Q3: How does it compare with other data?
- Q4: How well do we forecast future patterns?

Dataset description We performed experiments on the following three real datasets.

- *MemeTracker*: This dataset covers three months of blog activity from August 1 to October 31 2008², It contains short quoted textual phrases (“memes”), each of which consists of the number of mentions over time. We choose 1,000 phrases in blogs with the highest volume in a 7-day window around their peak volume.
- *Twitter*: We used more than 7 million Twitter³ posts covering an 8-month period from June 2011 to January 2012. We selected the 10,000 most frequently used hashtags.
- *GoogleTrends*: This dataset consists of the volume of searches for various queries (i.e., words) on Google⁴. Each query represents the search volumes that are related to keywords over time.

10.4.1 Q1: Explaining K-SC clusters

The results on this dataset were already presented in section 10.1 (see Figure 10.1). Our model correctly captures the six patterns of K-SC. Table 10.3 gives a further description of the SPIKEM fitting. Our model consists of seven parameters, each of which describes the behavior of spikes. Note that the total populations N are almost the same for all patterns, (around 2,000 to 3,000). This is because these six patterns are scaled on the y-axis so that they all have a peak volume of 100. We can see that $\beta * N$ is between 0.7 – 1.0 for these six patterns. We also see that Pattern C3 has an extreme shock $S_b = 114$ at time $n_b = 40$, which means that this spike is strongly affected by the external burst of activity (see Figure 10.1 (c)). On the other hand, Patterns C4-C6 have several peaks about 24 hours apart with a strength $P_a \simeq 0.4$.

We also evaluated our fitting accuracy by using the root mean square error (RMSE) between estimated values and real values: $RMSE = \sqrt{\frac{1}{n_d} \sum_n^{n_d} (X(n) - \Delta B(n))^2}$. Table 10.4 shows the fitting accuracy result for six patterns of K-SC. We compared SPIKEM with SI model. As discussed in section 10.3 (see Figure 10.2), SI cannot model the tail parts of the spikes. On the other hand, our solution, SPIKEM achieves high accuracy for every pattern of K-SC.

²<http://memetracker.org/>

³<http://twitter.com/>

⁴<http://www.google.com/insights/search/>

	C1	C2	C3	C4	C5	C6
N	2407	1283	1466	3079	4183	3435
$\beta * N$	0.95	1.00	0.86	0.92	0.79	0.69
n_b	26	17	40	35	0	34
S_b	4.73	0.06	114.13	23.24	2.58	45.58
ϵ	0.36	0.01	0.43	1.48	0.32	13.97
P_a	0.18	0.06	0.22	0.38	0.28	0.39
P_s	12	5	7	6	2	2

Table 10.3: The model parameters of our SPIKEM best fitting on six patterns of K-SC (see Figure 10.1).

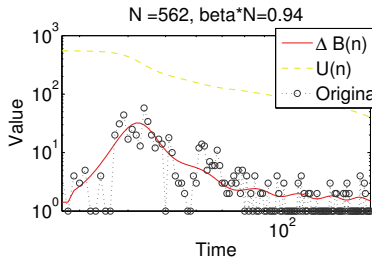
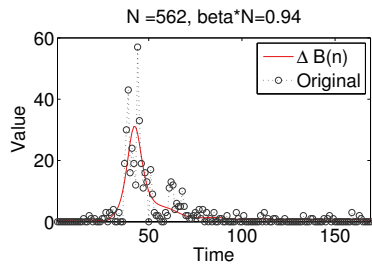
Pattern	C1	C2	C3	C4	C5	C6
SPIKEM	1.84	1.61	0.97	4.08	3.33	5.89
SI	15.64	6.78	19.65	25.29	20.36	21.76

Table 10.4: Fitting accuracy of SI vs. SPIKEM on six patterns of K-SC. SPIKEM consistently outperforms SI with respect to accuracy (RMSE) between the original values and the models.

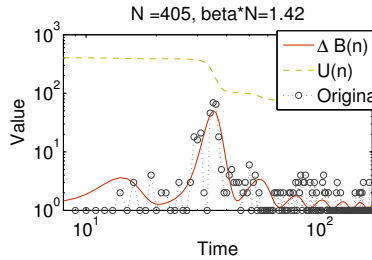
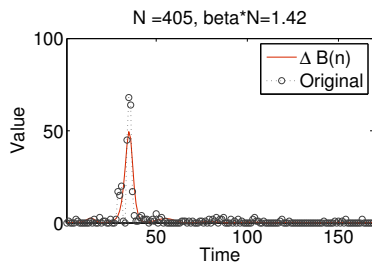
10.4.2 Q2: Matching *MemeTracker* patterns

Figure 10.3 shows the results of model fitting on the *MemeTracker* dataset. We selected six typical sequences according to the K-SC clusters. That is, each sequence corresponds to each pattern (C1-C6). We show the original sequences (black dots) and SPIKEM fitting, $\Delta B(n)$ (red line) in both linear-linear (top) and log-log (bottom) scales. In the log-log scale, we also show the count of un-informed bloggers, $U(n)$. In Figure 10.3, the bottom table shows the short phrases (memes) of each sequence. All of the phrases are sourced from U.S. politics in 2008. We obtained several observations for each sequence:

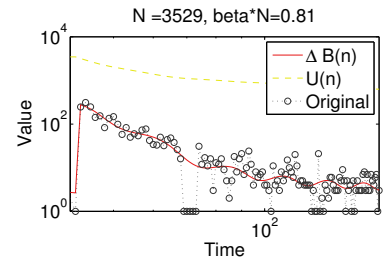
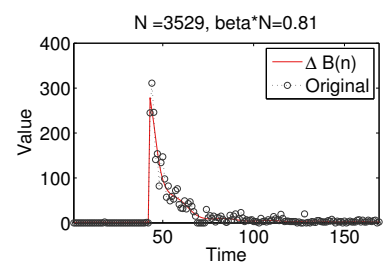
- Patterns C1 and C2: almost the same size of population, $N \simeq 500$, except that C2 has a quicker rise and fall (i.e., stronger infection, $\beta * N = 1.4$) than C1 ($\beta * N = 0.94$).
- Pattern C3: this sequence has a sudden rise and a power law decay. There is a slight daily periodicity.
- Patterns C4 and C5: there are clearly daily periodicities. Pattern C6, “lipstick on a pig” has the largest population of all six sequences (i.e., $N = 6259$).
- Pattern C6: the sequence: “yes we can” consists of huge spikes around $n = 40$, and constant periodic noise. This is because the bloggers mention this phrase as Barack Obama’s slogan as well as with more general meanings. We can also find that there are several extreme points (i.e., missing values) around $n = 120$ (see blue circle in log-log scale).



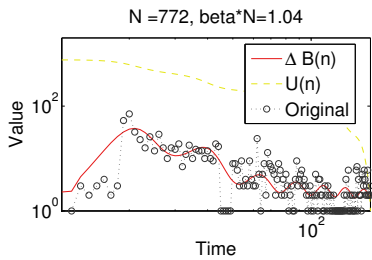
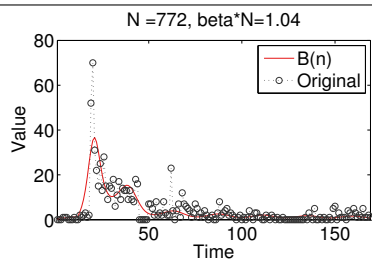
(a) Pattern C1: Meme #109



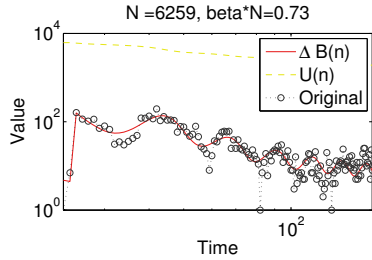
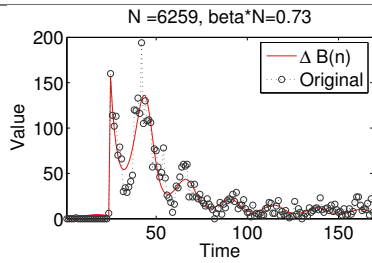
(b) Pattern C2: Meme #34



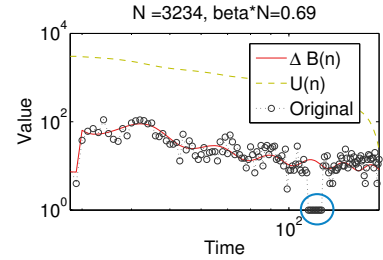
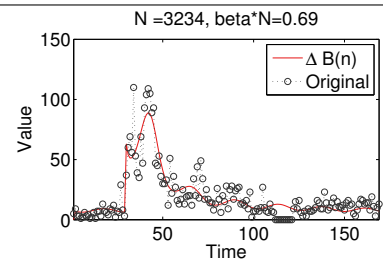
(c) Pattern C3: Meme #13



(d) Pattern C4: Meme #87



(e) Pattern C5: Meme #9



(f) Pattern C6: Meme #3

#109	the most serious financial crisis since the great depression	#87	what is required of us now is a new era of responsibility
#34	i love this country too much to let them take over another election	#9	you can put lipstick on a pig
#13	hope over fear, unity of purpose over conflict and discord	#3	yes we can yes we can

Figure 10.3: Results of SPIKEM fitting on six patterns from *MemeTracker* dataset. The figures show in both ‘linear-linear’ (top) and ‘log-log’ (bottom) scales. The bottom table lists the phrase (“meme”) of each patterns.

10.4.3 Q3: Matching other data

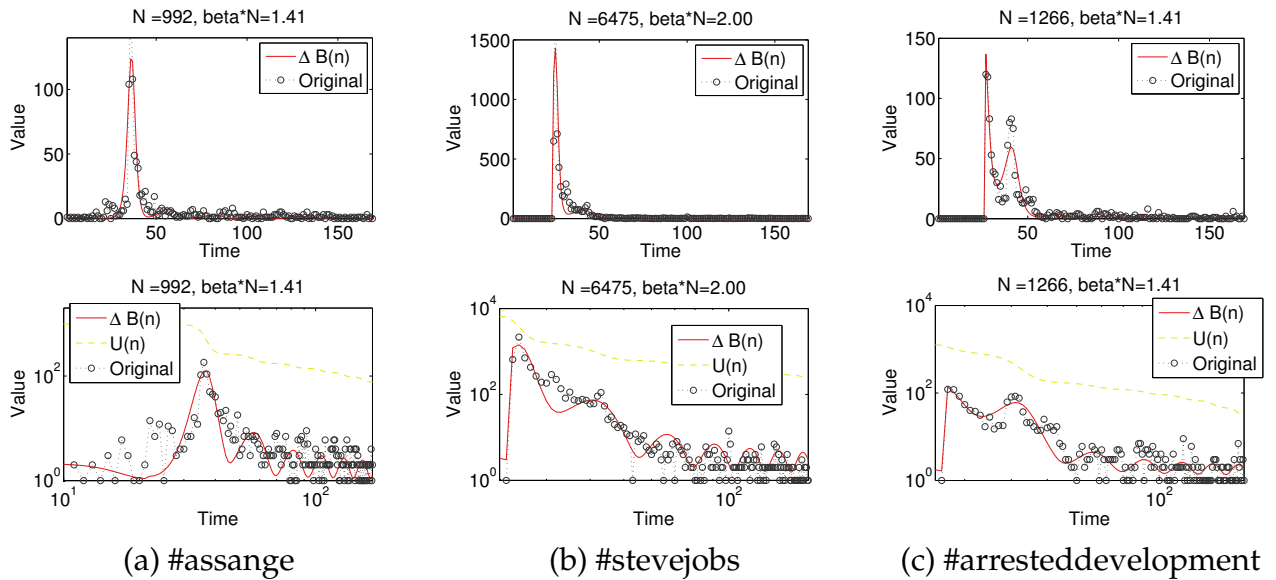


Figure 10.4: Results of SPIKEM fitting on three hashtags from *Twitter* dataset. The top and bottom rows show in linear-linear scale, and log-log scale, respectively.

We also demonstrate the effectiveness of our model for other types of spikes.

Fitting on Twitter data. Figure 10.4 describes our fitting results on the *hashtags* of *Twitter* data. In this figure, we can see that *Twitter* data behave similarly to *MemeTracker* data. Due to space limitations, we show only three major hashtags. Note that the top and bottom rows are in linear-linear and log-log scales, respectively. Our model captures the following characteristics: (a) *#assange*: this is a topic about Julian Assange, the founder of WikiLeaks. There are several mentions before the peak point (December 5, 2011). (b) *#stevejobs*: there is a sudden peak on October 5, 2011, with a long heavy tail (see Figure 10.4(b) in log-log scale). This was caused by the death of Steve Jobs. (c) *#arresteddevelopment*: this a topic about the movie “Arrested Development”. There is a clear daily periodicity with a peak point.

Fitting on GoogleTrend data. We can also observe influence propagation in queries on internet search engines. Figure 10.5 shows two different types of spikes on *GoogleTrends*. For an external catastrophic event (a) “tsunami”, we see that there is a super quick rise immediately after the Indian Ocean earthquake and tsunami in 2005. In contrast, (b) “harry potter” has a slower rise, which is because this spike was generated by “word-of-mouth” activity surrounding the release of a Harry Potter movie in 2007. SPIKEM evidently captures both types of spikes successfully.

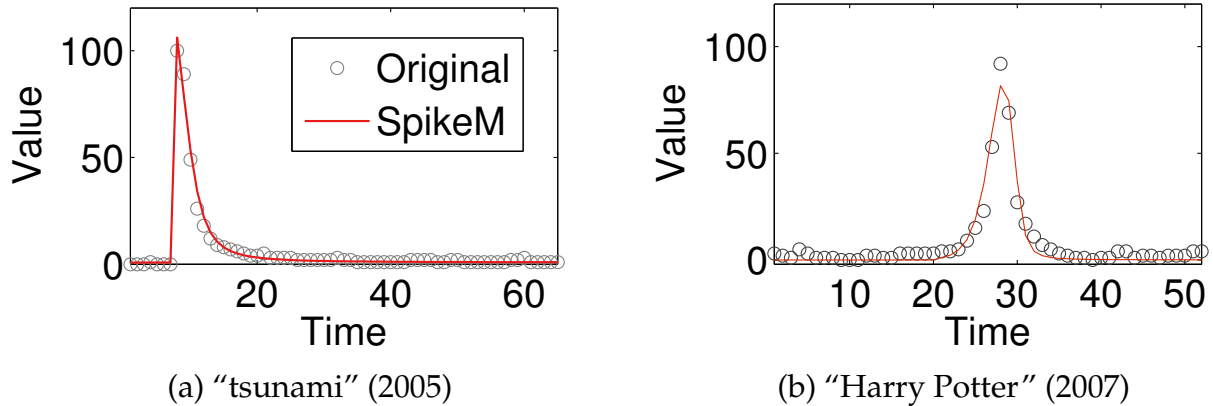


Figure 10.5: SPIKEM fitting on *GoogleTrends* dataset: the volume of searches for the keyword (in black dots) and fitting results (in red lines). Note that the window size is per week.

10.4.4 Q4: Tail-part forecasts

So far we have seen how SPIKEM captures the pattern dynamics for various spikes. Here, we answer a more practical question: given the first part of the spike, how can we forecast the future behavior of the tail part? Figure 10.6 shows results of our forecasts on *MemeTracker* data. We selected two the highest population phrases (#9 and #13 in Figure 10.3). We trained our models by using the values obtained over a period of 54 hours (solid black lines in the figure), and then forecasted the following days (solid red lines, about five days). Note that the vertical axis uses a logarithmic scale. We compared SPIKEM with the auto regressive model (AR). For a fair comparison, we used seven regression coefficients, which was the same size as our model parameters.

Our method achieves high forecasting accuracy while AR failed to forecast future patterns. More specifically, the reconstruction errors of SPIKEM are $RMSE = 9.26$ and 8.93 for #9 and #13, while AR has errors of 13.98 and 14.19 . Similar trends are observed in other phrases, however we omit the results due to space limitations. More importantly, our model can forecast the rise part of spikes as well as the tail part (discussed in Section 10.5).

10.5 Discussion - SPIKEM at work

Our proposed model, SPIKEM is capable of various applications. Here, we describe important applications and show some usefulness examples of our approach.

10.5.1 “What-if” forecasting

We have discussed tail-part forecasting in subsection 10.4.4. Ideally, we want to forecast not only the tail-part, but also the rise-part of a spike. This is much more difficult, because we usually have very few points in the rise-part of a spike. However, if this is a repeating

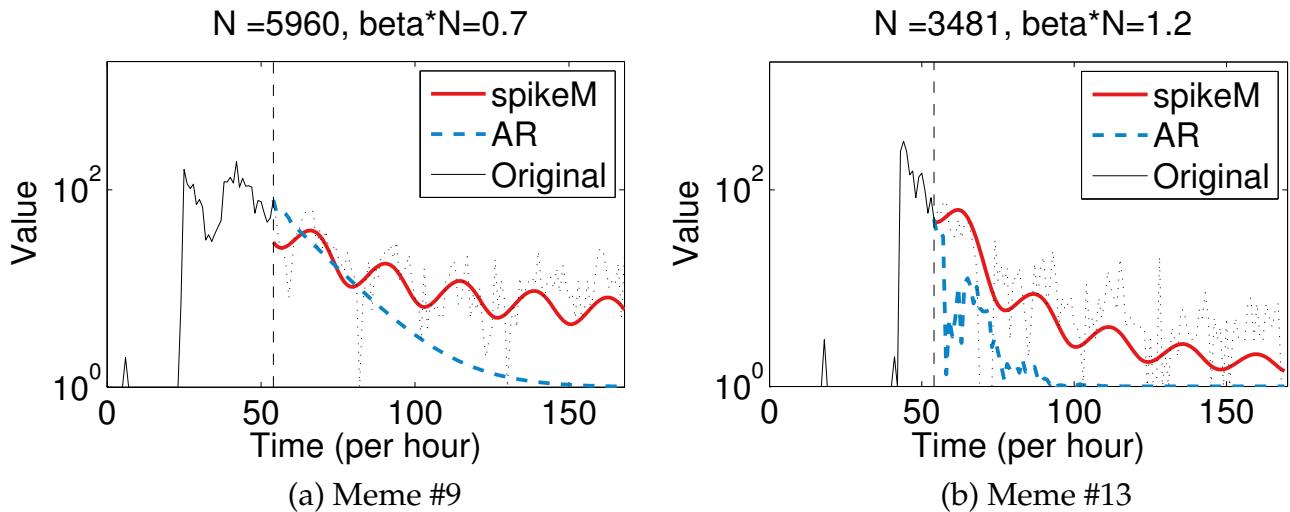


Figure 10.6: Results of tail-part forecasting on *MemeTracker* data. We train spikes from $n = 0$ to 54, and then, start forecasting at time $n = 54$. Our SPIKEM reflects reality better, while AR quickly converges to the zero.

event, like, say, the spikes induced by ‘Harry Potter’ movies releases, can we forecast future spikes if we know the release date of the next movie? It turns out that our SPIKEM model can help with this (difficult) task, too.

Thus, the problem we address in Figure 10.7 is as follows: we are given (a) the first spike in 2009, “Harry Potter and the Half-Blood Prince” ($n = 185$); (b) the release dates of the two sequel movies (blue text with as arrows pointed at $n = 255$ and 289), and (c) the access volume before the release dates (and specifically from 8 to 2 weeks before). Can we forecast the rise and fall shapes of upcoming spikes and their peak points?

Solution and results. SPIKEM can predict the potential population N of users who are interested in “Harry Potter”, and the strength of ‘word-of-mouth’ infection: β . Our solution is to assume that these values are fixed for all of the sequel spikes. The only difference is the strength of the “external shock”, i.e., n_b and S_b . Our solution consists of the following three-step process:

1. Train the parameter set θ by using the first spike (solid black line in the figure).
2. With the fixed parameters θ , infer the new values of \tilde{n}_b and \tilde{S}_b by using the beginning part of the next spike (blue lines between double arrows at $n = 250$ and 280).
3. Generate the spikes using θ and \tilde{n}_b and \tilde{S}_b (red lines).

In conclusion, Figure 10.7 shows that our model successfully captures the two sequel spikes and peak points n_{mode} .

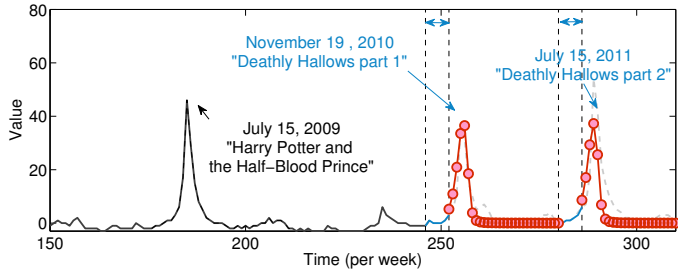


Figure 10.7: Results of “what-if” forecasting for the Harry Potter series. We trained parameters by using (a) the first spike around July 15, 2009 (black solid line), and (b) access volume two months before the release (blue lines with double arrows around time $n = 250, 280$) and then, forecasted the following two spikes (red lines).

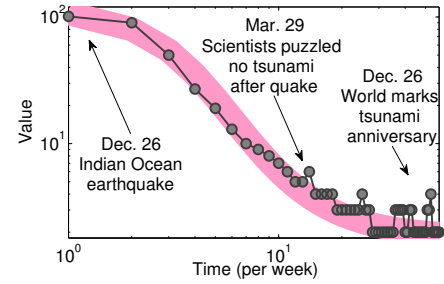


Figure 10.8: Outlier detection on *GoogleTrends* dataset (in log-log scale). Notice that the biggest spike, “world marks tsunami anniversary” occurred after one year (i.e., 52 weeks later).

10.5.2 Outlier detection

Since SPIKEM has a very high fitting accuracy on real datasets (described in section 10.4), another natural application would be anomaly detection. Figure 10.8 shows the fitting result of Figure 10.5 (a), in a **log-log** scale. Note that the black circles are the original sequence, and the pink line is our model fitting. We can visually observe that there are several points that do not overlap the model. For example, (a) on March 29, there is one spike, since another earthquake occurred on March 28. (b) There is a huge spike on December 26, 2005, which is exactly one year after the Indian Ocean earthquake.

10.5.3 Reverse engineering

Most importantly, our model can provide an intuitive explanation such as the potential number of interested bloggers, and the quality of news. Here we report our discoveries on *MemeTracker* and *Twitter* datasets (see Figure 10.9).

Observation 10.1 (Total population of bloggers). *The total populations of potential bloggers/users N are almost same for both datasets (around $N = 1,000 - 2,000$).*

We also note that they are skewed to the right, i.e., there is a long tail of larger values.

Observation 10.2 (Strength of first infection). *The strength of the “first burst” is $\beta * N \simeq 1.0$ for each dataset.*

The above two observations agree with the intuition: we can see common behavior for *MemeTracker* and *Twitter*, which means that they have similar characteristics in terms of social activities.

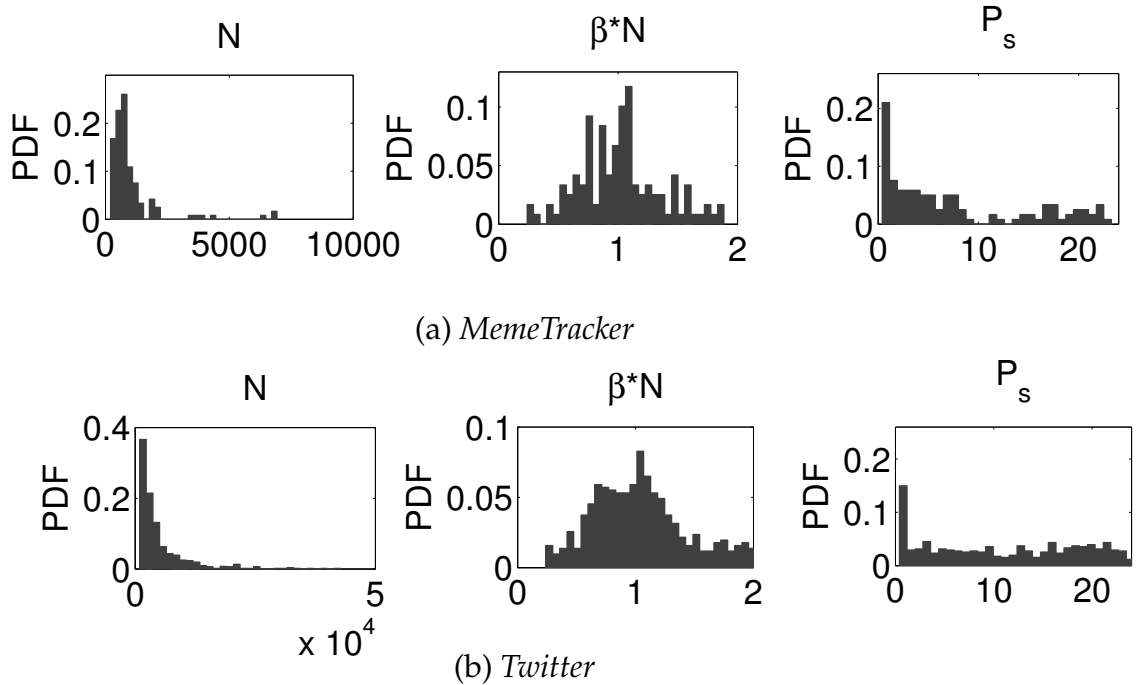


Figure 10.9: Reverse engineering: pdf of three parameters: $N, \beta * N, P_s$ over 1,000 memes/hashtags. (a) *MemeTracker*: total potential bloggers $N \simeq 1,000$, and strength of “first burst” $\beta * N \simeq 1.0$. More than 90% of the memes have clear daily periodicity with high activities around 6pm (i.e., $P_s \simeq 0$). (b) *Twitter*: similar trends except more spread in P_s , possibly, due to multiple time zone. Also see the text for more observations.

Observation 10.3 (Common activity and periodicity). *Typical user behavior is to have a daily periodicity with (a) phase shift $P_s = 0$ (small population during early morning, large population at peak point, 6pm) for MemeTracker, while (b) more spread in P_s .*

Note that more than 90% of all spikes have a daily periodicity in both datasets. The only the difference between the two datasets is that *Twitter* has several P_s values. This is because *Twitter* has multiple time zones (e.g., US, UK, Australia, and India).

10.6 Related Work

We present the related work, in three areas: time series analysis, influence propagation, and burst detection.

Time series Analysis. This is an old topic, that has attracted huge interest, and that is dealt with in well-regarded textbooks [BJR94]. Traditional approaches applied to data mining include Auto-Regression (AR) and variations [LLL⁺11], or Linear dynamical systems (LDS), Kalman filters (KF) and variants [JCW04, LPF10?] but they are all linear methods. Non-linear methods for forecasting tend to be hard to interpret, because they rely on nearest-neighbor search [CF02], or artificial neural networks [WG94]. Similarity search, indexing and pattern discovery in time sequences have also attracted huge interest

[FRM94, KS01, GKMS01, KPZ⁺04, LKL⁺04, VKY09, PAP⁺11, SFY07, SPF05, MSY09], but none of these methods specifically focused on modeling bursts.

Influence propagation. The canonical text-book for epidemiological models like SI is Anderson and May [AM91]. The power-law decay of influence has been reported in blogs [MLF⁺07], with a exponent of -1.5. Barabasi and his colleagues reported exponents of -1 and -1.5, for the response time in correspondence [Bar05]. Analyses of epidemics, blogs, social media, propagation and the cascades they create have attracted much interest [LBK09, YL10, KMM10, PCF⁺11, PBRF12, KKT03, TPT⁺10, GLMF09, GLNGT04, GKRT04, LAH06], and recently the reverse problem ('find who started it') [LTGM10, SZ11].

Burst detection. Remotely related to our work are the efforts to spot bursts. This includes the work of Kleinberg [Kle02], the algorithm of Zhu and Shasha [ZS03], and the algorithm of Parikh et al. [PS08]. None of the above gives a parsimonious model for describing the activity in a network.

10.7 Conclusions

We studied the rise-and-fall patterns in information diffusion process through online medias in this chapter. We presented SPIKEM, a general, accurate and succinct model that explains the rise-and-fall patterns. Our proposed SPIKEM has the following appealing advantages:

- **Unification power:** it includes earlier patterns and models as special cases (K-SC, as well as the SI model);
- **Practicality:** it matches the behavior of numerous, diverse, real datasets, including the power-law decay and much more beyond;
- **Parsimony:** our model requires only a handful of parameters;
- **Usefulness:** we showed how to use our model to do 'short-term' forecasting, to answer what-if scenarios, to spot outliers, and to learn more about the mechanisms of the spikes.

Chapter 11

Patterns amongst Competing Tasks

If Alice has double the friends of Bob, will she also have double the phone-calls (or wall-postings, or tweets)? We show that the answer to the question is a power-law: sub-linear, or super-linear, for a wide variety of diverse settings: tasks in a phone-call network, like count of friends, count of phone-calls, total count of minutes; tasks in a twitter-like network, like count of tweets, count of followees etc.

Why are there so many super-linear relationships? We answer this question through our proposed “competing tasks” model (CTM), that leads exactly to power-law relationships between task-frequencies. In a nutshell, the harder the task, the exponentially less frequently it will happen. Our model is inspired from population ecology, competing viruses (Chapter 4), using a generalization of the famous Volterra-Lotka prey-predator model. We illustrate our observations on two large, real datasets, spanning $\sim 2.2\text{M}$ and $\sim 3.1\text{M}$ individuals with 5 features each. We show how to use our observations to spot clusters and outliers, like, e.g., telemarketers in our phone-call network.

11.1 Introduction

If ‘Alice’ has $n_A=50$ contacts and did $n_B=100$ phone-calls to them, what should we expect for ‘Bob’, who has twice the contacts? One would expect a linear relationship (double the contacts, double the phone-calls). However, we show that in numerous settings, the relationship is a power law, being sub- or super-linear.

The questions we want to answer here are the following:

1. **Q1: Why:** what is the reason that super-linear relationships are so prevalent?
2. **Q2: Practical use:** Can we answer ‘what-if’ scenarios? Can we find anomalies?

Our contributions are exactly the answers to the above questions:

- **A1:** We propose the CTM model inspired from population ecology, with tasks competing for resources (person’s time). Then, we show that the relative frequency of tasks is *exponentially* related to the task’s difficulty.

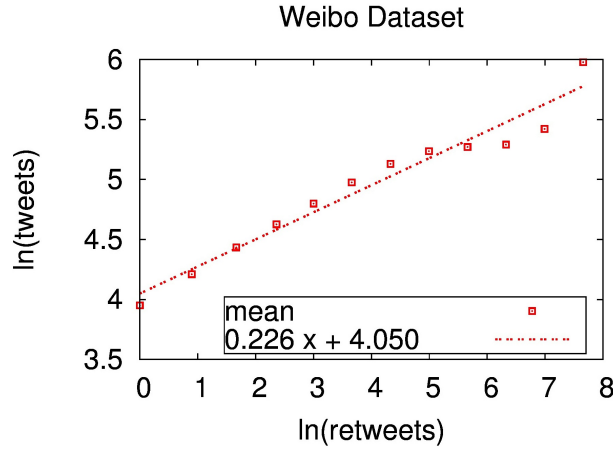


Figure 11.1: Illustration of super-linearity: Power-law relationship between count of tweets and count of retweets for each user in the Tencent-Weibo dataset. Log-log scales, each red square is the conditional average of tweets, for the given count of retweets. The last few points are noisy, because of extreme-value effects.

- **A2:** We show that our SURF fits several, diverse real datasets, and present how to spot outliers, and how to answer what-if questions.

Table 11.1: Symbols and definitions

Symbol	Definition
d_A, d_B	duration of task 'A' & 'B'
r_A, r_B	reproduction rate of species A' & 'B' - inverse of duration
n_A, n_B	population of species 'A' & 'B' = occurrence freq. of A' & 'B'
SURF	Super-Linear Relative Frequency Observation
CTM	Our Competing Tasks Model

We report results from two large, real, diverse datasets. The first spans ~ 3.1 M users and is on a phone-call dataset; for each customer, we study the count of distinct contacts, the count of phonecalls, the total minutes, and the count of text messages. The second is from Tencent-Weibo, a Chinese version of TwitterTM, with count of tweets, re-tweets, followees etc per user. Figure 11.1 illustrates our main idea: it depicts the power-law relationship between count of tweets and count of retweets. Both axis are logarithmic; each red square is the conditional average of tweets, for the given count of retweets. The last few points are noisy, because of extreme-value effects (there are very few people with so many re-tweets, and they dominate the average).

Table 11.1 gives the major symbols we use and their definitions. The rest of the sections are organized in the typical way: Proposed explanation and CTM Model; Experiments; Related work and Conclusions.

11.2 Competing Tasks Model (CTM)

Here we focus on question Q1 of the introduction - why do we see a power-law relationship between, say, tweets and retweets (Figure 11.1)?

We do a lot of tasks during each day. Let n_A and n_B be the frequencies of tasks 'A' and 'B' respectively, where, say, 'A' stands for 'going-for-a-walk' and 'B' for 'clicking-on-a-web-link'. Let d_A and d_B be the effort (say, time duration) that each task requires. One might expect that if a task takes twice as long, then it should happen twice less often: $n_A : n_B \stackrel{?}{=} d_B : d_A$.

However, this does not seem to agree with the numerous observations of the disproportionate impact of small changes in the effort to do any task. For example, the huge difference in organ donorship between culturally similar Germany and Austria (12% vs 99.9%), has been attributed to the difference in the 'default' choices - "in Austria, the default choice is to be an organ donor, whereas in Germany the default is not to be" [Wat11]. Similarly, the British physicist Stephen Hawking, notes in his famous popular science book 'A Brief History of Time', that "an editor warned him that for every equation in the book the readership would be halved" [Haw88]. Also, if a web form has 'opt-in' by default, the vast majority of people will opt-in, despite the fact that it only takes a few seconds to find the box and click the mouse. We shall refer to this observation as the RELATIVE EFFORT observation:

Observation 11.1 (RELATIVE EFFORT). *A small change in the effort (d_A) required for a task, makes a big difference in n_A , the number of times (frequency) we execute this task.*

Next we give a model that simultaneously (a) agrees with the intuitive RELATIVE EFFORT observation, and (b) also explains the super-linear relationship of frequencies n_A and n_B . We shall refer to the latter as SURF: *super-linear relative frequency* observation.

Specifically, we conjecture that the relative frequencies n_A and n_B follow a power-law, where the exponent depends on the relative effort (or time duration d_A and d_B) of the two tasks.

Conjecture 11.1 (Super-linear Relative Frequency (SURF)). *With n_A , n_B the task frequencies, and d_A , d_B the task durations, we have*

$$d_A * \log(n_A) = d_B * \log(n_B) = c \quad (11.1)$$

or equivalently

$$n_A = \exp(c/d_A) \quad (11.2)$$

where, c is some constant. This implies that a small change in the effort d_A of a task (eg., through a better web interface, requiring fewer clicks), will have an exponential impact on the frequency n_A of the task.

Empirical Evidence: Please see Figure 11.2 for numerous pairs of attributes of our real data, where the “super-linear relative frequency” observation (SURF) holds.

11.2.1 Justification

We can show that the conjecture holds, if we borrow the concept of competing species from population ecology: suppose we have two species of herbivores, ‘A’ and ‘B’, feeding on the same finite resource (say, R pounds of grass, on an isolated island). The species require d_A and d_B amounts of grass per individual to produce one offspring. How many members n_A and n_B of each species would we expect to see? This situation is equivalent to tasks ‘A’ and ‘B’, each competing for user’s time, and each requiring d_A and d_B units of time respectively.

We use the Lotka-Volterra-like equations [MM07] to model the competing species ‘A’ and ‘B’, with reproductive rate $r_A = 1/d_A$, $r_B = 1/d_B$, with d_A , d_B being the amount of resource (grass) needed to create offspring. Then, we have

$$\frac{\partial n_A}{\partial t} = r_A n_A (R - (d_A n_A + d_B n_B)) \quad (11.3)$$

$$\frac{\partial n_B}{\partial t} = r_B n_B (R - (d_A n_A + d_B n_B)) \quad (11.4)$$

where, $\frac{\partial n_A}{\partial t}$ is the time-derivative of the population of species A, $\frac{\partial n_B}{\partial t}$ is the time-derivative of the population of species B, and R is some maximum total units of resources for the species compete. The equations above state that the rate of change of a species, say A, is dependent on its reproductive rate (r_A), its current population (n_A), and the current resource units remaining ($R - (d_A n_A + d_B n_B)$). Given such a system of equations, we can now prove the following theorem:

Theorem 11.1 (SURF (Super-linear Relative Frequency)). *Under the setting described in Equations (11.3) and (11.4), we have*

$$n_A(t) \propto n_B(t)^{d_B/d_A} \quad (11.5)$$

where, $n_A(t)$ and $n_B(t)$ are the populations of the species at some time $t > 0$.

Proof. By dividing Equations (11.3) and (11.4), we obtain

$$\begin{aligned}
\frac{\partial n_A}{\partial n_B} &= \frac{r_A n_A}{r_B n_B} \\
\Rightarrow \int_{n_A(0)}^{n_A(t)} \frac{\partial n_A}{n_A} &= \frac{r_A}{r_B} \int_{n_B(0)}^{n_B(t)} \frac{\partial n_B}{n_B} \\
\Rightarrow \ln n_A(t) &= \ln n_B(t)^{\frac{r_A}{r_B}} + c \\
\Rightarrow n_A &= n_B^{r_A/r_B - c r_A} \\
\Rightarrow n_A(t) &\propto n_B(t)^{d_B/d_A}
\end{aligned}$$

where c is some constant and $n_A(0)$ and $n_B(0)$ are the initial populations of the species at time $t = 0$. □ □

Thus, the longer a task takes (d_A, d_B), the exponentially less often it will occur.

11.3 Experiments

In the following subsections we present the experimental results that corroborate our SURF theory. The real datasets we studied in order to check the validity of the proposed model are the following:

- Tencent Weibo (W)¹: Weibo is one of the largest micro-blogging websites in China. We studied the behavior of ~ 2.2 million users; for each one we extracted five quantities: the number of her tweets, retweets, comments, mentions and followees.
- Phonecall dataset (P): This dataset consists of phone-call records from a mobile provider in a big Asian city. For each of the ~ 3.1 million customers we obtained the number of her calls, messages, “voice” and “sms” friends, as well as the total minutes of her phone-calls.

The extracted features from each dataset correspond to the occurrences of the tasks (n_A, n_B etc.) in our SURF theory.

11.3.1 CTM at Work

In Fig. 11.2 we present the scatter plots of pairs of tasks that differ in the effort that they require.

Observation 11.2. *The SURF pattern holds in the real-world datasets; there is a power-law relationship between tasks of different difficulty.*

Here is a short explanation of the scatter plots: each user/customer is a blue point on the plane and is characterized by the number of times she did tasks A and B. Note that

¹Tencent Weibo Dataset, KDD Cup 2012. <http://www.kddcup2012.org>

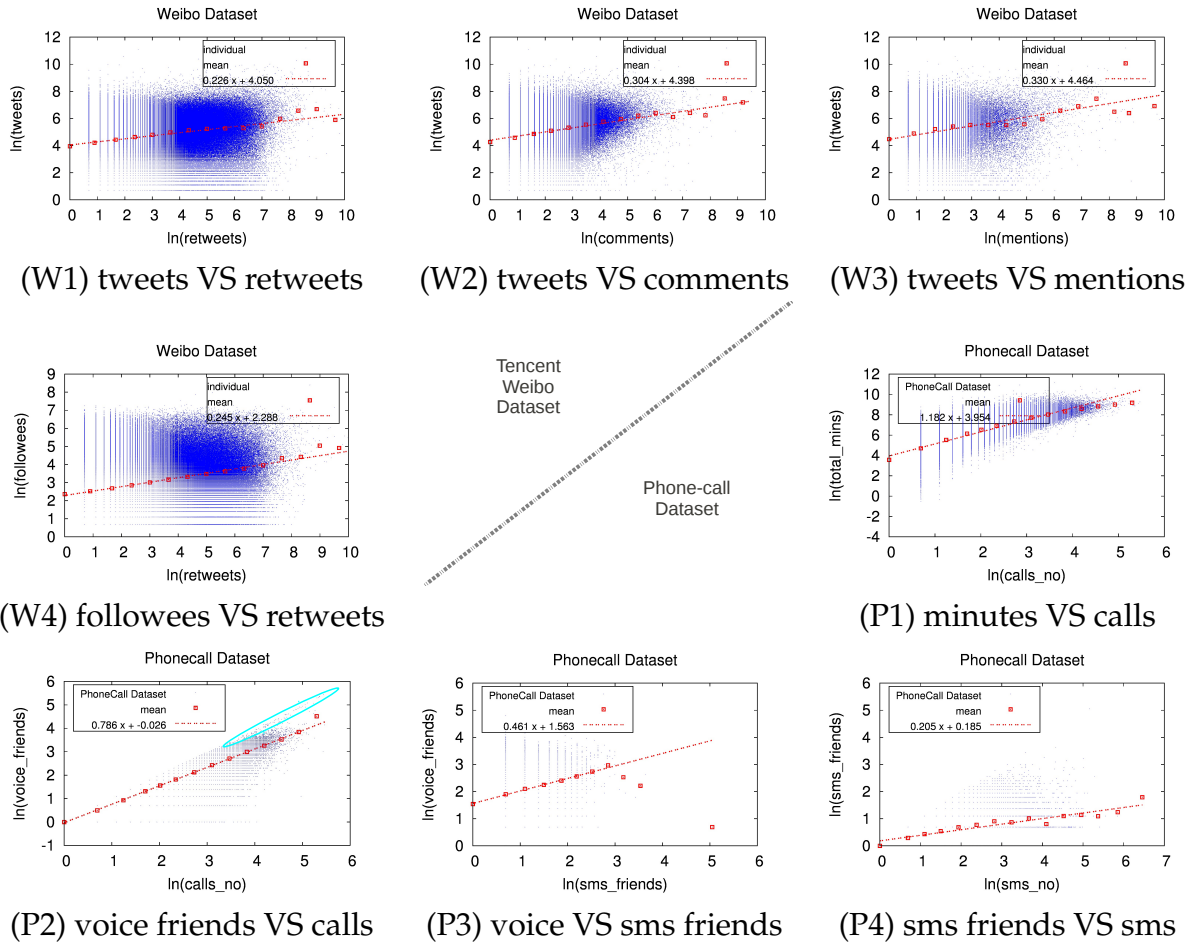


Figure 11.2: SURF patterns in real datasets (log-log scale): observe the power-law relationship between tasks of different difficulty. Note that for each dataset we have $n = 5$ features. We are giving the plots for $n - 1$ pairs, instead of $\binom{n}{2}$. (“W”: Tencent Weibo, and “P”: Phonecall data)

all plots “suffer” from heavy over-plotting, especially for small values of occurrences (not visible in the plots).

Linear regression in this cloud of points fails due to over-plotting, and so we resort to the following solution: we group the points in logarithmic buckets and compute the mean (red points) of Y given X . The line, $E[Y|X = x]$, is obtained by linear regression on the red points (ignoring the few last points, where the observations are extremely sparse, possibly due to the “horizon effect”). As we observe, in all cases the conditional expectation is a linear function of x .

Observation 11.3. *Deviations from the SURF patterns, as shown in Fig. 11.2(P2) are due to outliers, e.g. telemarketers.*

The customers within the cyan ellipse all have about 100 contacts, and 100 phone-calls, that is about one phone-call per contact. The rest of the population has many more phone-calls than contacts, leading to the suspicion that the former are telemarketers.

11.4 Related Work

Population Ecology: Population ecology is an area of its own, with numerous books and milestone papers [MM07]. The competing species analysis is discussed, among others, in [Ste09], along with the full extension of the famous Lotka-Volterra model for prey-predator competition. Recall that we analyzed (Chapter 4) a probabilistic competition model for competing diseases with ‘winner-takes-all’ phenomenon [PBRF12].

Power laws: Power laws have been discovered in numerous cases, often in conjunction with fractals and self-similarities [Sch91]. Some of the most famous power laws are the Zipf distribution [Zip49] and the Pareto distribution [Par96]. They have negative slopes, though. Power laws with positive slopes have also been discovered (length of coastlines, number of quad-tree blocks versus granularity [FG96]), and more recently in graphs: the number of edges grows super-linearly on the number of nodes [LKF05], with exponents 1.6 or 1.2; the number of participating triangles of a node grows super-linearly to the degree [Tso08], with exponent ~ 1.5 ; the total weight of a node is superlinear on the degree [MAF08], to name a few.

However, none of the above articles provided a solution to our setting, namely, an explanation for the super-linearity we observe, and validation on several, diverse datasets.

11.5 Conclusions

The contributions are the answers to the questions we posed in the introductions: Q1: why do we see super-linear relationships between counts of tasks; Q2: how can we put it to practical use. Specifically, the contributions are:

1. **A1:** Discovery and explanation of power law (CTM) in several, real, diverse datasets, on most of their n-choose-2 pairs of attributes/tasks.
2. **A2:** Illustration that SURF can be used for anomaly detection, clustering and what-if scenarios.

Future work may include developing more expressive models to account for the variance in the observations.

Part IV
Conclusion

Chapter 12

Conclusions and Future Directions

Networks are really ubiquitous, providing a simple yet powerful abstraction to tackle several real-world problems. In addition though, when considering large graphs, epidemics are also everywhere. For social networks, infectious diseases like the flu are prime examples, but hypes/memes are similarly epidemic in nature; whether it is friends discussing that latest gadget or phone, or sharing a funny video, there are nodes ‘infecting’ each other. Similarly, a computer virus can cause an epidemic in a computer network, as can a contaminant in a water distribution network. Applications include *cyber security*, *epidemiology* and *public health*, *product marketing* to *information dissemination*.

Many fundamental questions underlie the propagation-like processes in all these domains, a number of which we addressed in this thesis. Many of these questions are also inter-connected, and as this thesis demonstrates, answering some of them can be crucial for others. We tackled multiple important questions, like understanding *the tipping point* behavior of epidemics, predicting *who-wins* among competing viruses/products, developing effective algorithms for *immunization* and *marketing* for several real-world settings, and building more realistic online *information-diffusion models* while analyzing numerous real-datasets.

12.1 Summary of contributions

We summarize the major contributions and impact of this thesis next. Answering problems spanning multiple domains necessitates a multi-pronged approach. Hence, the contributions span three inter-dependent areas:

- I (Theory) *Discovering the importance of eigenvalues and winner-takes-all phenomena* (Chapters 2, 3, 4, 5)
- II (Algorithms) *Orders of magnitude faster and substantially more effective algorithms for immunization, and culprits detection* (Chapters 6, 7, 8, 9)
- III (Models) *More expressive, unifying, interpretable and predictive models using numerous real-world datasets* (Chapters 10, 11)

We are arguably the *first* to present a systematic study of propagation and immunization of *single* as well as *multiple* viruses on *arbitrary, real* and *time-varying* networks as the vast majority of the literature focuses on structured topologies, cliques, and related un-realistic models.

Theory

- **Eigenvalues for Epidemic Threshold:** We showed that the epidemic threshold for single viruses depends on the largest eigenvalue of an ‘appropriate’ matrix for *arbitrary* static and time-varying graphs. Our eigenvalue result generalizes and unifies previous results and has broad implications and applications like faster epidemiological simulations. We are the **first** to show the threshold on arbitrary static graphs and almost *any* virus propagation model. In addition, ours is the **first** closed formula for *any* set of arbitrary time-varying graphs.
- **Winner-Takes-All for Competing Viruses:** We are the **first** to show the surprising ‘winner-takes-all’ phenomenon involving competing viruses/products, spreading on *arbitrary* graphs. Additionally, we are the **first** to extend this problem to mutually-interacting viruses and show a phase-transition.
- ★ Impact: Our paper on epidemic thresholds on static graphs [PCF⁺11] was selected for one of the best papers of the conference. Our results have been used to enable other important tasks like anomaly detection and graph modeling [AMF10], and immunization (see Part II of this thesis). They are also being incorporated into *FRED*, an epidemiological simulator developed by MIDAS.

Algorithms

- **Dramatically better Immunization:** Our carefully designed algorithms such as NETSHIELD and SMART-ALLOC substantially improved the state-of-the-art in solving the complete and fractional immunization problems respectively. NETSHIELD outperformed many methods by more than **7 orders** of magnitude in running time, and competitors like the well-known acquaintance immunization in quality of solutions. Using task-specific structure SMART-ALLOC achieves up to **6x fewer** infections and **30,000x speed-up**, over current practice and ad-hoc heuristics on real hospital patient-transfer networks like US-MEDICARE and PENN-ALL (the current practice has been largely focused within individual hospitals).
- **Parameter-free Culprits detection:** Our algorithm NETSLEUTH is the **first linear-time** algorithm (in edges and nodes) for **automatically** identifying the set of culprits which best describes a given snapshot of the epidemic (both in identity and number)—in contrast to the state of the art (which are at least quadratic in running time, and require the number of seeds as input).
- ★ Impact: Our results and algorithms have been incorporated into undergraduate courses (UPitt Summer Program) and slides sought-after for graduate courses

(Xifeng Yan, UCSB) in universities, and have appeared in ACM Crossroads.

Models

- **Unifying models for online diffusion:** We developed SPIKEM, a powerful succinct model to explain the rise and fall patterns of information diffusion which **unifies** and includes earlier patterns and models, matches behavior of numerous real datasets and can be used to **forecast**, answer ‘**what-if**’ scenarios and even *reverse-engineer* epidemics.
- **Explaining power-laws in competing tasks:** We developed CTM, an intuitive model to explain the prevalence of **super-linear** relationships between the frequencies of various competing tasks observed in real-datasets, and use it to spot outliers like **telemarketers**.

12.2 Vision and Future directions

Looking ahead, the long-term goal is to solve large real-world problems by building models and understanding the dynamics of networked systems, be it social, technological, or natural. To solve high-impact problems involving complex systems, our vision is a *multi-pronged* ‘end-to-end’ research pipeline: (a) defining and collecting rich network structure (collaborating with domain experts), (b) understanding network properties and then abstracting/analyzing real-life processes on these networks, and (c) developing and implementing efficient algorithms to manipulate such processes for our benefit.

This thesis has made several major steps in these broad directions—we now better understand the effect of graph topology on various propagation processes (like epidemic thresholds), have better algorithms for many immunization and marketing tasks (like SMART-ALLOC), and have better models for describing propagation scenarios (like SPIKEM). Several challenges remain, some of which we describe next.

12.2.1 Long Term Challenges

Scalability: One of the major thrust areas would be continuing focusing on big-data and computing systems. We have already used data-intensive map-reduce type architectures (Hadoop) and distributed compute-intensive systems (Condor) for our current research. Such systems would not only allow us to perform large-scale analysis and develop deployable solutions for real problems, but may also serve as a source of interesting problems, e.g., how do failures cascade in Hadoop clusters?

Richer settings & dynamic parameters: What happens when the meme/virus evolves over time, e.g, a flu-like virus becomes a mumps-like virus? What changes, if this happens in an adversarial manner, in a game-theoretic fashion? We studied what happens

when the underlying network changes with time in this thesis. We wish to further study changes in other parameters like the evolution of viruses and memes and effect of sudden introduction of other viruses/topics/products into the ecosystem. Similarly, how do node attributes, like gender and age, affect the distribution of vaccines? How important are weak social ties (as opposed to strong friendship) in dynamical phenomena? How does additional uncertainty in edges change our answer to the culprits problem? Most existing models and algorithms work on plain graphs, where all nodes and edges are the same. Incorporating richer attributed network data—having auxiliary and uncertain features like historical attributes, textual information, geographical information, can greatly enhance models as well as algorithms.

Finally, we believe our *inter-disciplinary* approach is also vital here and has indeed led to many discoveries in this thesis, spanning areas like public health, social media and networking. We also need to use tools from Data Mining, Machine Learning, Statistical Physics and Biology to build, and analyze the models and mechanisms. And all of these have to be done in the context that we are trying to both *understand* and *manage* real-life processes which are on a societal-scale. These are pretty exciting times for research in networks.

Bibliography

- [AA05] Azmy Ackleh and Linda Allen. Competitive exclusion in sis and sir epidemic models with total cross immunity and density-dependent host mortality. *Discrete and Continuous Dynamical Systems-Series B*, 5, 2005. 4.2, 4.6
- [AJB00] R. Albert, H. Jeong, and A.-L. Barabási. Error and attack tolerance of complex networks. *Nature*, 407(6794):378–482, July 2000. 2.7.1
- [AM82] R. M. Anderson and R. M. May. Coevolution of hosts and parasites. *Parasitology*, 85, 1982. 4.2, 4.6
- [AM91] Roy M. Anderson and Robert M. May. *Infectious Diseases of Humans*. Oxford University Press, 1991. 2.2.1, 2.4, 2.5.2, 3.1, 3.2, 5, 1, 7.2, 7.3, 9.2.2, 10.6
- [AMF10] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. OddBall: Spotting Anomalies in Weighted Graphs. In *PAKDD*, 2010. 1.2, 12.1
- [Bai75] Norman Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. Griffin, London, 1975. 3.2, 7.2
- [Bar05] Albert L. Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 435, 2005. 10.3.1, 10.6
- [Bas69] Frank M. Bass. A new product growth for model consumer durables. *Management Science*, 15(5):215–227, 1969. 2.1, 2.2.2, 10.1
- [BB05] Hannah Brückner and Peter Bearman. After the promise: the std consequences of adolescent virginity pledges. *Journal of Adolescent Health*, 2005. 5.6.2
- [BBE⁺08] Christopher L. Barrett, Keith R. Bisset, Stephen G. Eubank, Xizhou Feng, and Madhav V. Marathe. Episimdemics: an efficient algorithm for simulating the spread of infectious disease over large realistic social networks. pages 1–12, 2008. 2.8.3, 2.9, 3, 3.1
- [BBV10] A. Barrat, M. Barthélemy, and A. Vespignani. *Dynamical Processes on Complex Networks*. Cambridge University Press, 2010. 2.4
- [BHW92] Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. A theory of fads, fashion, custom, and cultural change in informational cascades. *Journal of Political Economy*, 100(5):992–1026, October 1992. 2.2.2, 7.2

- [BJR94] George E.P. Box, Gwilym M. Jenkins, and Gregory C. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, Englewood Cliffs, NJ, 3rd edition, 1994. 10.6
- [BKS07] S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. *WINE*, 2007. 4.2
- [BLP03] Linda Briesemeister, Patric Lincoln, and Philip Porras. Epidemic profiles and defense of scale-free networks. *WORM 2003*, Oct. 27 2003. 6.2, 7.2
- [BPP⁺10] Sergey V. Buldyrev, Roni Parshani, Gerald Paul, H. Eugene Stanley, and Shlomo Havlin. Catastrophic Cascade of Failures in Interdependent Networks. *Nature*, 464(7291):1025–1028, April 2010. 2.2.3
- [BPRF12] Alex Beutel, B. Aditya Prakash, Roni Rosenfeld, and Christos Faloutsos. Interacting viruses in networks: can both survive? In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '12*, pages 426–434, 2012. 1.1.2
- [BS11] A. N. Bishop and I. Shames. Link operations for slowing the spread of disease in complex networks. *EPL*, 95(1), 2011. 8.6
- [CCF⁺10] Caroline Colijn, Ted Cohen, Christophe Fraser, William Hanage, Edward Goldstein, Noga Givon-Lavi, Ron Dagan, and Marc Lipsitch. What is the mechanism for persistent coexistence of drug-susceptible and drug-resistant strains of streptococcus pneumoniae? *Journal of The Royal Society Interface*, 7(47):905–919, 2010. 5.1
- [CCHL96] Carlos Castillo-Chavez, Wenzhang Huang, and Jia Li. Competitive exclusion in gonorrhea models and other sexually transmitted diseases. *SIAM J. Appl. Math*, 56, 1996. 4.2, 4.6
- [CCHL99] Carlos Castillo-Chavez, Wenzhang Huang, and Jia Li. Competitive exclusion and coexistence of multiple strains in an sis std model. *SIAM J. Appl. Math*, 59, 1999. 4.2, 4.6
- [CDS98] Dragos M. Cvetković, Michael Doob, and Horst Sachs. *Spectra of Graphs: Theory and Applications, 3rd Revised and Enlarged Edition*. Vch Verlagsgesellschaft Mbh, December 1998. 9.4.3
- [CF02] Deepay Chakrabarti and Christos Faloutsos. F4: Large-scale automated forecasting using fractals. *CIKM 2002*, November 2002. 10.6
- [CHbA03] Reuven Cohen, Shlomo Havlin, and Daniel ben Avraham. Efficient immunization strategies for computer networks and populations. *Physical Review Letters*, 91(24), December 2003. 2.8.1, 6.2, 6.7.2, 6.8.2, 7.2
- [CLF⁺07] Deepayan Chakrabarti, Jure Leskovec, Christos Faloutsos, Samuel Madden, Carlos Guestrin, and Michalis Faloutsos. Information Survival Threshold in Sensor and P2P Networks. In *IEEE INFOCOM*, 2007. 2.2.3

- [CLV03] Fan Chung, Linyuan Lu, and Van Vu. Eigenvalues of random power law graphs. *Annals of Combinatorics*, 7(1), 2003. 2.7.1
- [CPMF04] D. Chakrabarti, Spiros Papadimitriou, D. S. Modha, and Christos Faloutsos. Fully automatic cross-associations. In *Proceedings of the 10th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, Seattle, WA, pages 79–88, 2004. 9.6
- [CPS10] Claudio Castellano and Romualdo Pastor-Satorras. Thresholds for epidemic spreading in networks. *Phys. Rev. Lett.*, 105, December 2010. 2.2.1
- [CS08] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. In *PNAS*, 2008. 10.1, 10.1, 10.2
- [CT06] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience New York, 2006. 9.3.2
- [CV05] Rudi Cilibrasi and Paul Vitányi. Clustering by compression. *IEEE Transactions on Information Technology*, 51(4):1523–1545, 2005. 9.6
- [CWW⁺08] D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos. Epidemic thresholds in real networks. *ACM TISSEC*, 10(4), 2008. 2.1, 2.2.1, 3.1, 3.2, 3.1, 3.5, 4.4.4, 5.6.1, 6.4.2
- [CWW10] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. *KDD*, 2010. 2.2.2, 9.6
- [DLJ08] Chris H. Q. Ding, Tao Li, and Michael I. Jordan. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *ICDM*, pages 183–192, 2008. 6.2
- [DW04] Peter S. Dodds and Duccan J. Watts. A generalized model of social and biological contagion. *Journal of Theoretical Biology*, 232:587–604, September 2004. 2.1
- [DWG10] T. Donker, J. Wallinga, and H. Grundmann. Patient referral patterns and the spread of hospital-acquired infections through national health care networks. *PLoS Comput Biology*, 6(3):e1000715, 2010. 7.1
- [EGAK⁺04] Stephen Eubank, Hasan Guclu, V. S. Anil Kumar, Madhav V. Marathe, Aravind Srinivasan, Zoltan Toroczkai, and Nan Wang. Modelling disease outbreaks in realistic urban social networks. *Nature*, 429(6988):180–184, May 2004. 2, 2
- [EK10] D. Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010. 2.1, 2.4, 2.5.2
- [EPL09] N. Eagle, A. Pentland, and D. Lazer. Inferring social network structure using mobile phone data. *Proc. of the National Academy of Sciences*, 106(36), 2009. 3.6, 6.8.3

- [FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. *SIGCOMM*, pages 251–262, Aug-Sept. 1999. 2.2.1
- [FG96] Christos Faloutsos and Volker Gaede. Analysis of the z-ordering method using the hausdorff fractal dimension. *VLDB*, September 1996. 11.4
- [Fie73] M. Fiedler. Algebraic connectivity of graphs. 1973. 6.2
- [FM07] Christos Faloutsos and Vasilis Megalooikonomou. On data mining, compression and Kolmogorov complexity. In *Data Mining and Knowledge Discovery*, volume 15, pages 3–20. Springer-Verlag, 2007. 9.6
- [Fre77] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977. 6.2, 6.7.2, 8.6
- [Fre79] L. C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1:215–239, 1979. 7.1
- [FRM94] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. In *Proc. ACM SIGMOD*, pages 419–429, Minneapolis, MN, May 25-27 1994. 10.6
- [GGLNT04] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW '04*, 2004. 2.2.2, 7.2
- [GJ83] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1983. 7.3.2
- [GKMS01] Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, and Martin Strauss. Surfing wavelets on streams: One-pass summaries for approximate aggregate queries. In *VLDB*, pages 79–88, 2001. 10.6
- [GKRT04] R. Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of trust and distrust. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004. ACM. 10.6
- [GL96] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996. 8.3.2
- [GLL11] Amit Goyal, Wei Lu, and Laks V. S. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. *ICDM*, 2011. 9.6
- [GLM01] Jacob Goldenberg, Barak Libai, and Eitan Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 2001. 2.2.2, 7.2
- [GLMF09] Michaela Goetz, Jure Leskovec, Mary McGlohon, and Christos Faloutsos. Modeling blog dynamics. In *ICWSM*, 2009. 10.6
- [GLNGT04] D. Gruhl, David Liben-Nowell, R. Guha, and A. Tomkins. Information

- diffusion through blogspace. *SIGKDD Explor. Newsl.*, 6(2):43–52, December 2004. 10.6
- [GMT05] Ayalvadi Ganesh, Laurent Massoulié, and Don Towsley. The effect of network topology on the spread of epidemics. In *IEEE INFOCOM*, Los Alamitos, CA, 2005. IEEE Computer Society Press. 2.2.1, 3.2, 5.6.1, 7.2, 7.3.1
- [Gr7] Peter Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007. 9.1, 9.2.3, 9.6
- [Gra78] M. Granovetter. Threshold models of collective behavior. *Am. Journal of Sociology*, 83(6):1420–1443, 1978. 2.1, 2.2.2, 4.2, 7.2
- [GRLK10] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '10*, pages 1019–1028, New York, NY, USA, 2010. ACM. 2.2.2
- [GVL89] G. H. Golub and C. F. Van-Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 2nd edition, 1989. 3.4.2, 7.5.2
- [Haw88] Stephen Hawking. *A Brief History of Time*. Bantam Dell Publishing, 1988. 11.2
- [HBW11] Habiba and Tanya Berger-Wolf. Working for influence: effect of network density and modularity on diffusion in networks. *ICDM DaMNet*, 2011. 9.6
- [Het00] H. W. Hethcote. The mathematics of infectious diseases. *SIAM Review*, 42, 2000. 2.1, 2.2.1, 2.4, 2.4, 2.5.2, 2.2, 3.1, 3.2, 3.3, 4.3.1, 5, 5.3.1, 1
- [HJ91] Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991. 2.E.1, 2
- [HKYH02] Petter Holme, Beom Jun Kim, Chang No Yoon, and Seung Kee Han. Attack vulnerability of complex networks. *Phys. Rev. E*, 2002. 8.6
- [HMM03] Yukio Hayashi, Masato Minoura, and Jun Matsukubo. Recoverable prevalence in growing scale-free networks and the effective immunization. *arXiv:cond-mat/0305549 v2*, Aug. 6 2003. 2.1, 6.2, 7.2
- [HO74] A. G. Hawkes and D. Oakes. A cluster representation of a self-exciting process. *J. Appl. Prob.*, 11:493–503, 1974. 10.2
- [HS74] M. W. Hirsch and S. Smale. *Differential Equations, Dynamical Systems and Linear Algebra*. Academic Press, 1974. 2.2, 3.1, 4.4.1, 2
- [HT08] Nagamochi H. and Ibaraki T. *Algorithmic Aspects of Graph Connectivity*. Cambridge University Press, 2008. 6.2, 6.4.1
- [HW97] C. He and G. A. Watson. An algorithm for computing the numerical radius. *IMA Journal of Numerical Analysis*, 17, 1997. 2

- [HY84] H. W. Hethcote and J. A. Yorke. Gonorrhea transmission dynamics and control. *Springer Lecture Notes in Biomathematics*, 46, 1984. 2.1, 2.4
- [ICM⁺09] T. J. Iwashyna, J. D. Christie, J. Moody, J. M. Kahn, and D. A. Asch. The structure of critical care transfer networks. *Medical Care*, 47(7):787–793, 2009. 7.1, 7.2
- [IW02] Eitan Israeli and R. Kevin Wood. Shortest-path network interdiction. *Networks*, 40(2):97–111, 2002. 8.6
- [JCW04] Ankur Jain, Edward Y. Chang, and Yuan-Fang Wang. Adaptive stream resource management using kalman filters. In *SIGMOD*, pages 11–22, 2004. 10.6
- [Kar72] Richard M. Karp. *Reducibility Among Combinatorial Problems*. New York, 1972. 8.3.1
- [KG07] Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. In *AAAI*, pages 1650–1654, 2007. 6.6.3
- [KKT03] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD '03*, 2003. 2.1, 2.2.2, 4.2, 7.2, 9.6, 10.6
- [Kle98] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. In *ACM-SIAM Symposium on Discrete Algorithms*, 1998. 6.2, 6, 7.4.2, 8.6
- [Kle02] Jon M. Kleinberg. Bursty and hierarchical structure in streams. In *KDD*, pages 91–101, 2002. 10.6
- [Kle07] Jon Kleinberg. The wireless epidemic. *Nature*, Vol. 449, Sep 2007. 2.1, 2.4
- [KMM10] Ravi Kumar, Mohammad Mahdian, and Mary McGlohon. Dynamics of conversations. In *SIGKDD*, pages 553–562, 2010. 10.6
- [KMST10] Alexandra Kolla, Yury Makarychev, Amin Saberi, and Shang-Hua Teng. Subgraph sparsification and nearly optimal ultrasparsifiers. In *STOC*, pages 57–66, 2010. 8.6
- [KNRT03] Ravi Kumar, Jasmine Novak, Prabhakar Raghavan, and Andrew Tomkins. On the bursty evolution of blogspace. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 568–576, New York, NY, USA, 2003. ACM Press. 2.2.2, 7.2
- [KOW08] J. Kosta, Y. A. Oswald, and R. Wattenhofer. Word of mouth: Rumor dissemination in social networks. *15 Intl. Coll. on Struct. Inform. and Comm. Complexity SIROCO*, 2008. 4.2
- [KPST11] U. Kang, Spiros Papadimitriou, Jimeng Sun, and Hanghang Tong. Centralities in large networks: Algorithms and observations. In *SDM*, pages 119–130, 2011. 8.6
- [KPZ⁺04] Eamonn J. Keogh, Themis Palpanas, Victor B. Zordan, Dimitrios Gunopulos, and Marc Cardle. Indexing large human-motion databases. In *VLDB*, pages

780–791, 2004. 10.6

- [KS01] Tamer Kahveci and Ambuj K. Singh. An efficient index structure for string databases. In *Proceedings of VLDB*, pages 351–360, September 2001. 10.6
- [KW93] J. O. Kephart and S. R. White. Measuring and modeling computer virus prevalence. *IEEE Computer Society Symposium on Research in Security and Privacy*, 1993. 2.1, 2.2.1, 2.4, 7.2
- [LAH06] Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. The dynamics of viral marketing. In *EC '06: Proceedings of the 7th ACM conference on Electronic commerce*, pages 228–237, New York, NY, USA, 2006. ACM Press. 2.2.2, 7.2, 10.6
- [LBK09] Jure Leskovec, Lars Backstrom, and Jon M. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, pages 497–506, 2009. 2.1, 10.1, 10.6
- [LCC⁺09] Marc Lipsitch, Caroline Colijn, Ted Cohen, William P. Hanage, and Christophe Fraser. No coexistence for free: Neutral null models for multi-strain pathogens. *Epidemics*, 1(1):2 – 13, 2009. 5.1, 5.2
- [Lev44] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathematics*, II(2):164–168, 1944. 10.3.3
- [Liv03] D. M. Livermore. Bacterial resistance: Origins, epidemiology, and impact. *Clinical Infectious Diseases*, 36(S1), 2003. 7.1
- [LKF05] Jure Leskovec, Jon M. Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD*, pages 177–187, 2005. 11.4
- [LKG⁺07] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie S. Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007. 6.2, 9.6
- [LKL⁺04] Jessica Lin, Eamonn J. Keogh, Stefano Lonardi, Jeffrey P. Lankford, and Donna M. Nystrom. Visually mining and monitoring massive time series. In *KDD*, pages 460–469, 2004. 10.6
- [LLL⁺11] Lei Li, Chieh-Jan Mike Liang, Jie Liu, Suman Nath, Andreas Terzis, and Christos Faloutsos. Thermocast: A cyber-physical forecasting model for data centers. In *KDD*, 2011. 10.6
- [LMF⁺07] Jure Leskovec, Mary McGlohon, Christos Faloutsos, Natalie S. Glance, and Matthew Hurst. Patterns of cascading behavior in large blog graphs. In *SDM*, 2007. 10.3.1
- [LPF10] Lei Li, B. Aditya Prakash, and Christos Faloutsos. Parsimonious linear fingerprinting for time series. *Proc. VLDB Endow.*, 3:385–396, September 2010. 10.6

- [LSH00] Weifa Liang, Xiaojun Shen, and Qing Hu. Finding the most vital edge for graph minimization problems on meshes and hypercubes. *International Journal of Parallel and Distributed Systems and Networks*, 2000. 8.6
- [LSN96] Marc Lipsitch, Steven Siller, and Martin A. Nowak. The evolution of virulence in pathogens with vertical and horizontal transmission. *Evolution*, 50, 1996. 5.1
- [LTGM10] Theodoros Lappas, Evimaria Terzi, Dimitrios Gunopulos, and Heikki Mannila. Finding effectors in social networks. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, Washington, DC, pages 1059–1068, 2010. 2.2.2, 7.2, 9.1, 9.4.5, 9.5.1, 9.6, 10.6
- [LV93] M. Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, 1993. 9.2.3
- [LZZ⁺03] M. Lad, X. Zhao, B. Zhang, D. Massey, and L. Zhang. Analysis of BGP Update Surge during Slammer Worm Attack. In *5th International Workshop on Distributed Computing (IWDC)*, 2003. 1
- [MAF08] Mary McGlohon, Leman Akoglu, and Christos Faloutsos. Weighted graphs and disconnected components: patterns and a generator. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 524–532, Las Vegas, Nevada, USA, 2008. 11.4
- [McC00] C. R. McCuler. The many proofs and applications of perron’s theorem. *SIAM Review*, 42, 2000. 2.F.2, 3.4.1, 4.4.4, 7.3.2, 7.5.1, 9.4.3
- [McK25] A G McKendrick. Applications of mathematics to medical problems. In *Proceedings of Edin. Math. Society*, volume 44, pages 98–130, 1925. 2.2.1, 7.2
- [Mea12] Paul R. McAdam and et. al. Molecular tracing of the emergence, adaptation, and transmission of hospital-associated methicillin-resistant staphylococcus aureus. *Proc. of Natl. Acad. of Sciences of USA (Early Edition)*, 2012. 7.1
- [Mil63] Harold Willis Milnes. Conditions that the zeros of a polynomial lie in the interval $[-1, 1]$ when all zeros are real. *The American Mathematical Monthly*, 70, No. 7, Aug. - Sept. 1963. 2.F.2
- [MK09] Jose Marcelino and Marcus Kaiser. Reducing influenza spreading over the airline network. *PLoS*, 2009. 8.6
- [MKC⁺04] Nilly Madar, Tomer Kalisky, Reuven Cohen, Daniel ben Avraham, and Shlomo Havlin. Immunization and epidemic dynamics in complex networks. *Eur. Phys. J. B*, 38(2):269–276, 2004. 6.2
- [MLF⁺07] Mary McGlohon, Jure Leskovec, Christos Faloutsos, Matthew Hurst, and Natalie Glance. Finding patterns in blog shapes and blog evolution. In *International Conference on Weblogs and Social Media*, Boulder, Colo., March 2007. 10.6

- [MM02] Petar Maymounkov and David Mazières. Kademia: A Peer-to-Peer Information System Based on the XOR Metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems, IPTPS '01*, pages 53–65, London, UK, 2002. Springer-Verlag. 2.8.4
- [MM07] Robert May and Angela McLean. *Theoretical Ecology: Principles and Applications*. Oxford University Press, 3rd edition, 2007. 11.2.1, 11.4
- [MNP06] Lauren Ancel Meyers, M.E.J. Newman, and Babak Pourbohloul. Predicting epidemics on directed contact networks. *Journal of Theoretical Biology*, 240(3):400 – 418, 2006. 7.2
- [MSN10] Attilio Milanese, Jie Sun, and Takashi Nishikawa. Approximating spectral impact of structural perturbations in large networks. *Phys. Rev. E*, 81, 2010. 3
- [MSP⁺12] Yasuko Matsubara, Yasushi Sakurai, B. Aditya Prakash, Lei Li, and Christos Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '12*, pages 6–14, 2012. 1.1.4
- [MSY09] Yasuko Matsubara, Yasushi Sakurai, and Masatoshi Yoshikawa. Scalable algorithms for distribution search. In *ICDM*, pages 347–356, 2009. 10.6
- [MW03] James Moody and Douglas R. White. Social cohesion and embeddedness: A hierarchical conception of social groups. *American Sociological Review*, pages 1–25, 2003. 6.2, 8.6
- [MW08] J. Ian Munro and Dorothea Wagner. Better approximation of betweenness centrality. 2008. 6.2
- [NDS07] NDSSL. Synthetic Data Products for Societal Infrastructures and Protopopulations: Data Set 2.0. *NDSSL-TR-07-003*, 2007. 2, 2
- [New02] M. E. J. Newman. Spread of epidemic disease on networks. *Phys. Rev. E*, 66(1):016128, Jul 2002. 2.2.1
- [New05a] Mark E. J. Newman. Threshold effects for two pathogens spreading on a network. *Physical Review Letters*, 95(10):108701, September 2005. 2.2.1, 2.9
- [New05b] M.E.J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27:39–54, 2005. 6.2, 6.7.2, 8.6
- [NJW01] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, pages 849–856, 2001. 6.2
- [PAP⁺11] Panagiotis Papapetrou, Vassilis Athitsos, Michalis Potamias, George Kollios, and Dimitrios Gunopulos. Embedding-based subsequence matching in time-series databases. *ACM Trans. Database Syst.*, 36(3):17, 2011. 10.6
- [Par96] V. Pareto. *Oeuvres Completes*. Droz, Geneva, 1896. 11.4
- [PBG11] Manos Papagelis, Francesco Bonchi, and Aristides Gionis. Suggesting ghost

- edges for a smaller world. In *CIKM*, pages 2305–2308, 2011. 8.6
- [PBMW98] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998. Paper SIDL-WP-1999-0120 (version of 11/11/1999). 6.2, 6.7.2, 8.6
- [PBRF12] B. Aditya Prakash, Alex Beutel, Roni Rosenfeld, and Christos Faloutsos. Winner takes all: Competing products or ideas on fair-play networks. pages 1037–1046, 2012. 2.2.2, 5.5.2, 10.6, 11.4
- [PBS10] Nishith Pathak, Arindam Banerjee, and Jaideep Srivastava. A generalized linear threshold model for multiple cascades. *ICDM*, 2010. 2.2.2, 4.2
- [PCF⁺11] B. Aditya Prakash, Deepayan Chakrabarti, Michalis Faloutsos, Nicholas Valler, and Christos Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. In *ICDM*, 2011. 1.2, 4.4.4, 4.6, 5.6.1, 6.4.2, 6.7.2, 7.2, 7.3.1, 10.6, 12.1
- [Phi93] Cynthia A. Phillips. The network inhibition problem. In *STOC*, pages 776–785, 1993. 8.6
- [PS08] Nish Parikh and Neel Sundaresan. Scalable and near real-time burst detection from ecommerce queries. In *KDD*, pages 972–980, 2008. 10.6
- [PSV01] R. Pastor-Satorras and A. Vespignani. Epidemic spreading in scale-free networks. *Physical Review Letters* 86, 14, 2001. 2.1, 2.2.1, 2.4, 2.7.1
- [PSV02] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic dynamics in finite size scale-free networks. *Physical Review E*, 65:035108, 2002. 3.1, 7.2
- [PTV⁺10] B. Aditya Prakash, Hanghang Tong, Nicholas Valler, Michalis Faloutsos, and Christos Faloutsos. Virus propagation on time-varying networks: Theory and immunization algorithms. *ECML-PKDD*, 2010. 2.9
- [PVF12] B. Aditya Prakash, Jilles Vreeken, and Christos Faloutsos. Spotting culprits in epidemics: How many and which ones? In *ICDM*, 2012. 1.1.3
- [RD02] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing, 2002. 2.2.2, 7.2, 9.6
- [Ris83] Jorma Rissanen. Modeling by shortest data description. *The Annals of Statistics*, 11(2):416–431, 1983. 9.3.1
- [Rog03] Everett M. Rogers. *Diffusion of Innovations, 5th Edition*. Free Press, August 2003. 2.2.2, 3.1, 7.2
- [Sch91] Manfred Schroeder. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. W.H. Freeman and Company, New York, 1991. 11.4
- [SFY07] Yasushi Sakurai, Christos Faloutsos, and Masashi Yamamuro. Stream monitoring under the time warping distance. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, April 15-20, 2007, The*

Marmara Hotel, Istanbul, Turkey, pages 1046–1055, 2007. 10.6

- [She95] Hong Shen. Finding the k most vital edges with respect to minimum spanning tree. *Acta Informatica*, 36:405–424, 1995. 8.6
- [SKOM12] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. Efficient discovery of influential nodes for sis models in social networks. *Knowledge and Information Systems (KAIS)*, 30(3):613–635, 2012. 2.2.2
- [SM97] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. In *CVPR*, pages 731–737, 1997. 6.2
- [SMHH11] Christian M. Schneider, Tamara Mihaljev, Shlomo Havlin, and Hans J. Herrmann. Restraining epidemics by improving immunization strategies. *CoRR*, abs/1102.1929, 2011. 8.6
- [SPF05] Yasushi Sakurai, Spiros Papadimitriou, and Christos Faloutsos. BRAID: Stream mining through group lag correlations. In *SIGMOD Conference*, pages 599–610, Baltimore, MD, USA, 2005. 10.6
- [SQCF05] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *ICDM*, pages 418–425, 2005. 6.2, 6.7.2
- [SS90] G. W. Stewart and Ji-Guang Sun. *Matrix Perturbation Theory*. Academic Press, 1990. 6.5.2, 6.5.2, 8.3.3, 8.3.3
- [Ste09] M. Henry H. Stevens. *A Primer of Ecology with R*. Springer Press, 2009. 5, 1, 11.4
- [Str88] Gilbert Strang. *Linear Algebra and its Applications*. Harcourt Brace Jonanovich, San Diego, 3rd edition, 1988. 9.4.3
- [SZ10] Devavrat Shah and Tauhid Zaman. Detecting sources of computer viruses in networks: theory and experiment. In *Proceedings of the ACM International Conference on Performance Evaluation (SIGMETRICS)*, pages 203–214, 2010. 9.1, 9.5.1, 9.6
- [SZ11] Devavrat Shah and Tauhid Zaman. Rumors in a network: Who’s the culprit? *IEEE Transactions on Information Technology*, 57(8):5163–5181, 2011. 9.1, 9.3.3, 9.4.5, 9.5.1, 9.6, 10.6
- [TPT⁺10] Hanghang Tong, B. Aditya Prakash, Charalampos E. Tsourakakis, Tina Eliassi-Rad, Christos Faloutsos, and Duen Horng Chau. On the vulnerability of large graphs. In *ICDM*, 2010. 7.2, 8.5.2, 8.8, 10.6
- [Tso08] Charalampos E. Tsourakakis. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *ICDM*, pages 608–617, 2008. 11.4
- [TTL05] Douglas Thain, Todd Tannenbaum, and Miron Livny. Distributed computing in practice: the condor experience. *Concurrency - Practice and Experience*,

- 17(2-4):323–356, 2005. 7.6.1
- [VGKG08] Milan Vojnovic, Varun Gupta, Thomas Karagiannis, and Christos Gkantsidis. Sampling strategies for epidemic-style information dissemination. *IEEE INFOCOM*, 2008. 2.1
- [VKY09] Michail Vlachos, Suleyman Serdar Kozat, and Philip S. Yu. Optimal distance bounds on time-series data. In *SDM*, pages 109–120, 2009. 10.6
- [VV04] N.K. Vereshchagin and P.M.B. Vitanyi. Kolmogorov’s structure functions and model selection. *IEEE Transactions on Information Technology*, 50(12):3265–3290, 2004. 9.3.1
- [VvS11] Jilles Vreeken, Matthijs van Leeuwen, and Arno Siebes. KRIMP: Mining itemsets that compress. *Data Mining and Knowledge Discovery*, 23(1):169–214, 2011. 9.6
- [Wat11] Duncan Watts. *Everything is obvious: Once you know the answer*. Random House, 2011. 11.2
- [WCWF03] Yang Wang, Deepayan Chakrabarti, Chenxi Wang, and Christos Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. In *Symposium on Reliable Distributed Systems*, pages 25–34, Los Alamitos, CA, 2003. IEEE Computer Society Press. 3.2, 7.2, 7.3.1, 8.5.2
- [(We11)] (Website). Mainline bittorrent website. November 15, 2011. 2.8.4
- [WG94] Andreas S. Weigend and Neil A. Gerschenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison Wesley, 1994. 10.6
- [Woo93] R. Kevin Wood. Network interdiction problem. *Mathematical and Computer Modeling*, 17(2):1–18, 1993. 8.6
- [WS98] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393:440–442, 1998. 2.7.1
- [YL10] Jaewon Yang and Jure Leskovec. Modeling information diffusion in implicit networks. In *ICDM*, pages 599–608, 2010. 10.6
- [YL11] Jaewon Yang and Jure Leskovec. Patterns of temporal variation in online media. In *WSDM*, pages 177–186, 2011. 10.1, 10.1, 10.1
- [Zac77] W. W. Zachary. An information flow model for conflict and fission in small groups. pages 452–473, 1977. 6.7.1
- [ZHD⁺01] Hongyuan Zha, Xiaofeng He, Chris H. Q. Ding, Ming Gu, and Horst D. Simon. Spectral relaxation for k-means clustering. In *NIPS*, pages 1057–1064, 2001. 6.2
- [ZIM⁺09] W. Zingg, A. Imhof, M. Maggiorini, R. Stocker, E. Keller, and C. Ruef. Impact of a prevention strategy targeting hand hygiene and catheter care on the incidence of catheter-related bloodstream infections. *Crit Care Med.*, 37(7):2167–2173, 2009. 7.1, 7.3, 7.6.1

- [Zip49] G.K. Zipf. *Human Behavior and Principle of Least Effort: An Introduction to Human Ecology*. Addison Wesley, Cambridge, Massachusetts, 1949. 11.4
- [ZS03] Yunyue Zhu and Dennis Shasha. Efficient elastic burst detection in data streams. In *KDD*, pages 336–345, 2003. 10.6
- [ZWF⁺11] Jichang Zhao, Junjie Wu, Xu Feng, Hui Xiong, and Ke Xu. Information propagation in online social networks: a tie-strength perspective. *Knowledge and Information Systems*, pages 1–20, 2011. 2.2.2