COMPUTATIONAL PERSPECTIVES ON LARGE-SCALE SOCIAL NETWORKS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Johan Holke Olof Ugander

August 2014

COMPUTATIONAL PERSPECTIVES ON LARGE-SCALE SOCIAL NETWORKS

Johan Holke Olof Ugander, Ph.D.

Cornell University 2014

This thesis investigates both how computational perspectives can improve our understanding of social networks, and also how modern insights about social networks can be put to work to address difficult computational and inferential challenges across systems engineering and the social sciences. The microstructure of human behavior has a rich history of study across many disciplines, and only recently — through the data deluge of online instrumentation and experimentation — has the role networks play across social and economic domains come into full view. Work in this thesis examines how social network neighborhoods, the rich local networks that surround individuals, function as contact surfaces through which individuals process information, mediating social decision and social contagion processes. Work in this thesis on distributing graph computations at Facebook, the online social networking service, has led to dramatic efficiency gains there, successfully deploying a new partitioning algorithm to reduce average query times for their "People You May Know" link prediction system by 50%. These improvements were achieved by harnessing both geographic and network structures of social graphs not necessarily found in other graph contexts. Additional work presents a highly scalable "restreaming" approach to partitioning massive graphs with rich local structure. Lastly, work on interference in online experiments (A/B tests) offers a framework based on graph partitioning to design lower variance estimators for treatment effects under network interference, a grand challenge of modern online experimentation. As individuals bring their social relations online, the web is rapidly evolving from a network of documents to a network of people, and computing with social data will require richer, novel methods for working with the subtleties that give social networks their distinctive character.

BIOGRAPHICAL SKETCH

Johan Ugander grew up in Woodcliff Lake, New Jersey and graduated from the Academy for Engineering and Design Technology (AEDT) in 2002, a program within the Bergen County Academies public magnet high school in Hackensack, New Jersey. After high school he spent two years in the deserts of eastern California attending Deep Springs College from 2002–2004. Born to expatriate Swedish parents, in 2004 he moved to Sweden and enrolled at Lund University, where he obtained an M.Sc. degree in Engineering Mathematics (*Civilingenjör Teknisk Matematik*) in 2008. The work for his masters thesis from Lund was completed at Caltech, as a visiting student in the Department of Control and Dynamical Systems. From 2008–2009 he studied at the University of Cambridge, United Kingdom, completing Part III of the Mathematical Tripos (CASM). In 2009 he enrolled as a Ph.D. student at the Center for Applied Mathematics at Cornell University in Ithaca, New York. During the summers of 2010, 2011, and 2012 he completed research internships at Facebook with their Data Science team. He spent the 2013–2014 academic year visiting the University of Washington. After graduating from Cornell he will complete a one year post-doctoral fellowship at Microsoft Research Redmond, before joining the faculty at Stanford University as Assistant Professor in the Management Science and Engineering Department in August 2015. He is an avid rock climber, and has promised himself to not let his research interfere too much with visits to Yosemite.

Dedicated to my parents:

"Problemet, mammi, är... Problemet, pappi, är..."

ACKNOWLEDGMENTS

First, I would like to thank my thesis advisor Jon Kleinberg for his unbelievably generous mentorship and friendship. I can not overstate my gratitude for his thoughtful guidance over the last five years. Second, I would like to thank Lars Backstrom, my host and collaborator at Facebook for much of the work in this thesis, as well as Cameron Marlow, Brian Karrer, and the entire Facebook Data Science team. Lars hosted me as an intern at Facebook during the summer of 2010, and it has been a tremendous privilege to work with him over the years. Cameron Marlow, who led the Data Science team throughout all my collaborations, is a rare combination of terrifically enthusiastic and carefully thoughtful, an inimitable research mentor. Brian Karrer was a terrific collaborator, and our joint effort assembling the "Anatomy of the Facebook Social Graph" paper [151], which is not part of this thesis, was a great privilege.

Chapter 2 of this thesis was a collaboration with Jon as well as Lars and Cameron at Facebook, and published in the Proceedings of the National Academy of Sciences [152]. Chapter 3 was a collaboration with Lars and Jon, and published in the Proceedings of the 2013 World Wide Web Conference (WWW) [154]. Chapter 4 was a collaboration with Lars, and published in the Proceedings of the 2013 ACM Conference on Web Search and Data Mining (WDSM) [153], where it was awarded the Best Student Paper Award. Chapter 5 was a collaboration with Joel Nishimura, a fellow Cornell Ph.D. student at the Center for Applied Mathematics, and was published at the 2013 ACM Conference on Knowledge Discovery and Data Mining (KDD) [122]. Chapter 6 was a collaboration with Jon as well as Brian and Lars at Facebook, and also published at the 2013 ACM Conference on Knowledge Discovery and Data Mining (KDD) [155]. I would like to thank Sinan Aral, Dean Eckles, James Fowler, Peter Grassberger, Isabel Kloumann, Michael Macy, Mathew Salganik, Cosma Shalizi, Steven Strogatz, Justin Vincent, and Duncan Watts, who all provided helpful comments on various drafts of various chapters

of this thesis, and who were all thanked in the acknowledgements sections of the various papers.

I have had the tremendous fortune of being mentored by many great individuals over the course of my education, starting at the Bergen County Academies with Dr. Bahadir Karuv, who advised the electronics research program, and Dr. Jon Grieco, who instilled in me at an early age the importance of seeking out great mentors. At Deep Springs I was very grateful to have Robley Williams and Jack Holt as scientific role models. At Lund I trained under many great educators, including Christian Söderberg, Gustaf Söderlind, as well as the entire Department of Automatic Control. I was very fortunate to have Mary Dunlop and Richard Murray at Caltech advise me first as a summer undergraduate SURF student and later for my Lund master's thesis. It was their guidance with regards to research process that planted the seeds that eventually led to this thesis. I would like to thank Einar Heiberg, who involved me in his medical software start-up Medviso during the summer of 2008, an invaluable experience. At Cambridge I was very fortunate to step in to Frank Kelly's "Stochastic Networks" course, and it was through my Part III essay under Frank's supervision that I transitioned from my early research interest in mathematical biology to more general theoretical questions about networks, rigorously formulated. At Cornell there were many faculty besides Jon who contributed significantly to my training, and I would like to thank Steven Strogatz, Dan Huttenlocher (both members of my thesis committee), Robert Kleinberg, and Éva Tardos in particular. During the last year visiting the University of Washington, I have been incredibly fortunate to have Carlos Guestrin host me and to serve as my deft career coach while navigating the academic job market.

Beyond my many exceptional senior mentors, this thesis would also not have been possible without the tremendous camaraderie that surrounded me at Cornell. During my time at Cornell I co-authored papers with fellow Ph.D. candidates Martin Larsson, Dan

Romero, Chenhao Tan, and Joel Nishimura, where the joint work with Joel Nishimura constitutes Chapter 5 of this thesis. Beyond co-authors, I would also like to thank the rest of the Center for Applied Mathematics (CAM) family, including in particular my officemates over the years: Michael Flashman, Matt Holden, Isabel Kloumann, Katie Montovan, and Elizabeth Wesson. Beyond my officemates, I often borrowed the ears of Diarmuid Cahalane, Scott Clark, Sarah Iams, Seth Marvel, and Tim Novikoff to help navigate various research questions. I would also like to thank the Cornell Networks Journal Club, a wonderful gathering of network enthusiasts across disciplines who were always energized to discuss networks. Outside the office, I am incredibly fortunate to have had as great apartmentmates as Christian Guzman, Scott Henderson, Sander Hunter, Martin Larsson, Karin Lilja, and Nimish Pujara. Martin and I were classmates in Lund who both enrolled in Ph.D. programs at Cornell. For the three years that we overlapped in Ithaca, Martin and I rarely ate less than five meals a week together. Most of these meals were lunches at Collegetown Bagels (CTB), and a number of them contributing to our 2011 NIPS paper [93], not a part of this thesis.

Lastly, I would like to thank my parents Margareta and Mikael, my siblings Martin and Olof, and my wife, Stephanie Safdi. My gratitude for my family's support is simply immeasurable. My parents provided me with a loving and supportive home where my passion for inquiry laid deep roots. Thank you for being such wonderful parents. My brothers Olof and Martin are my closest companions in life, and Martin's Ph.D. before mine inspired me from a young age that research was the good stuff, it was where the fun was to be had. I met my wife Stephanie shortly after arriving at Cornell, during my first climbing trip to the Shawnagunks in 2009. Her partnership and friendship is the only thing I've thought more about over the last five years than this thesis. I thank her for her boundless love, patient belays, support, and inspiration.

TABLE OF CONTENTS

Biographical Sketch . . . iii
Dedication . . . iv
Acknowledgments . . . v
Table of Contents . . . viii
List of Tables . . . xi
List of Figures . . . xii

1 Introduction **1**
1.1 Contributions . . . 9

2 Structural diversity in social contagion **15**
2.1 Introduction . . . 15
2.2 User recruitment . . . 17
2.3 User engagement . . . 22
2.4 Discussion . . . 27
2.5 Data collection . . . 28
2.5.1 Recruitment data collection . . . 28
2.5.2 Engagement data collection . . . 30
2.6 The k-brace . . . 30
2.7 Additional analyses of recruitment . . . 35
2.7.1 Five node neighborhood topologies . . . 35
2.7.2 Structural vs. demographic diversity . . . 35
2.7.3 Embeddedness and weak ties . . . 36
2.8 Additional analyses of engagement . . . 39
2.8.1 Predicted engagement for other neighborhood sizes . . . 39
2.8.2 Controlling for k-brace size . . . 39

3 Subgraph frequencies: mapping the empirical and extremal geography of large graph collections **41**
3.1 Introduction . . . 42
3.2 Data description . . . 46
3.3 Subgraph space . . . 47
3.4 Extremal bounds . . . 57
3.4.1 Background on subgraph frequency and homomorphism density 58
3.4.2 An LP for subgraph frequency bounds . . . 62
3.4.3 Bounding frequencies of arbitrary subgraphs . . . 65
3.5 Classification of audiences . . . 68
3.6 Conclusion . . . 72
3.7 Code . . . 74

4 Balanced label propagation for partitioning massive graphs 75
4.1 Introduction . . . 76
4.2 Balanced label propagation . . . 79
4.2.1 Partition constraints . . . 81
4.2.2 The constrained relocation problem . . . 81
4.2.3 Iteration . . . 85
4.2.4 Approximating utility gain . . . 85
4.3 Geographic initialization . . . 86
4.3.1 Balloon partitioning algorithm . . . 87
4.3.2 Oversharding . . . 90
4.4 Evaluating Performance . . . 91
4.4.1 Sharding the Facebook social graph . . . 91
4.4.2 LiveJournal comparison . . . 94
4.5 Realtime Deployment . . . 96
4.5.1 People You May Know . . . 96
4.5.2 Request concentration . . . 98
4.5.3 Query time and network traffic . . . 101
4.6 Discussion . . . 102

5 Restreaming graph partitioning: simple versatile algorithms for advanced balancing 104
5.1 Introduction . . . 105
5.2 Streaming partitioning . . . 108
5.2.1 The streaming model . . . 109
5.3 Restreaming Partitioning . . . 111
5.3.1 Restreaming LDG . . . 112
5.3.2 Restreaming FENNEL . . . 113
5.3.3 Convergence over restreams . . . 115
5.3.4 Dynamic graphs . . . 117
5.3.5 Parallelization . . . 118
5.4 Generalized types of balance . . . 118
5.4.1 Balancing other quantities . . . 119
5.4.2 Stratified balance . . . 120
5.5 Results . . . 123
5.5.1 Node balance results . . . 124
5.5.2 Tempering . . . 128
5.5.3 Other types of balance . . . 129
5.5.4 Stratified balance results . . . 129
5.5.5 Parallel results . . . 131
5.6 Conclusion . . . 132

6 Graph cluster randomization: network exposure to multiple universes 134
6.1 Introduction . . . 135
6.2 Network exposure models . . . 140
6.3 Randomization and estimation . . . 145
6.3.1 Exposure probabilities . . . 147
6.3.2 Estimator variance . . . 150
6.4 Variance on restricted-growth graphs . . . 153
6.4.1 Cycle and powers of the cycle examples . . . 154
6.4.2 Clustering restricted-growth graphs . . . 157
6.4.3 Variance bounds . . . 160
6.5 Conclusion . . . 162

7 Future work 164

Bibliography 166

LIST OF TABLES

3.1 Classification accuracy for Neighborhoods (N), Groups (G), and Events (E) on different sets of features. R_G and R_λ denote the residuals with respect to a $G_{n,p}$ and stochastic graph model baseline, as described in the text. 73

5.1 The percentage of edges cut (lower is better) for the basic methods studied in this work applied to a diverse collection of graphs partitioned into 40 different partitions. The restreamed methods were run for 10 restreaming iterations while the parallel versions were split across 30 workers and run for 30 restreaming iterations. METIS(1.03) is run with 3% slack, while METIS(1.001) is run with slack 0.1% slack. For each graph the best score excluding METIS(1.03) is bolded. 124

LIST OF FIGURES

2.1 Contact neighborhoods during recruitment. (a) An illustration of a small friendship neighborhood and a highlighted contact neighborhood consisting of four nodes and three components. (b–d) The relative conversion rates for two-node, three-node, and four-node contact neighborhood graphs. Shading indicates differences in component count. For five-node neighborhoods, see Figure 2.6. Invitation conversion rates are reported on a relative scale, where 1.0 signifies the conversion rate of one-node neighborhoods. Error bars represent 95% confidence intervals and implicitly reveal the relative frequency of the different topologies. 18

2.2 Recruitment contact neighborhoods and component structure. (a) Conversion as a function of edge count neighborhoods with one connected component (1 CC) with four to six nodes, where variations in edge count predict no meaningful difference in conversion. (b) Conversion as a function of neighborhood size, separated by CC count. When component count is controlled for, size is a negative indicator of conversion. (c) Conversion as a function of tie strength in two-node neighborhoods, measured by photo co-tags, a negative indicator of predicted conversion. Recruitment conversion rates are reported on a relative scale, where 1.0 signifies the conversion rate of one-node neighborhoods. Error bars represent 95% confidence intervals. 19

2.3 Inviter position during recruitment. Shown is recruitment conversion as a function of neighborhood graph topology and inviter position in neighborhoods of size 4. The position of the inviter within the neighborhood graph is described exactly (up to symmetries) by node degree. Shading indicates differences in component count. Recruitment conversion rates are reported on a relative scale, where 1.0 signifies the conversion rate of one-node neighborhoods. Error bars represent 95% confidence intervals. 21

2.4 Engagement and structural diversity for 50-node friendship neighborhoods. (a) Illustration of the connected components in a friendship neighborhood, delineating connected components and components of size ≥ 3. (b) Illustration of the k-core and the k-brace, delineating the connected components of the 2-core and the 1-brace. (c) Engagement as a function of connected component count. (d) Engagement as a function of the number of components of size $\geq k$, for $k = 2, 3, 4, 8$, with connected component (CC) count shown for comparison. (e) Engagement as a function of k-core component count for $k = 1, 2, 3$, with CC count shown for comparison. (f) Engagement as a function of k-brace component count for $k = 1, 2$, with CC count shown for comparison. Engagement rates are reported on a relative scale, where 1.0 signifies the average conversion rate of all 50-node neighborhoods. All error bars are 95% confidence intervals. For other neighborhood sizes, see Figure 2.9. 25

2.5 Engagement as a function of edge density. For five different neighborhood sizes, $n = 10, 20, 30, 40, 50$, we see that when component count is not accounted for, an internal engagement optimum is observed, showing the combined forces of focused context and structural heterogeneity. Engagement rates are reported on a relative scale, where 1.0 signifies the average conversion rate of all 50-node neighborhoods. All error bars are 95% confidence intervals. 26

2.6 Invitation conversion rates of size five neighborhoods, grouped by their number of connected components and their degree distribution, which equates to 31 equivalence classes (for graphs of size four or smaller, degree distribution is a unique determinant of isomorphism, but this is not true for graphs of size five or larger). For example, the label "1:[4,4,4,4,4]" indicates the clique, 1 component where all the nodes have degree 4. The label "2:[1,1,2,2,2]" indicates a graph of two components (a triangle and a pair), while the label "1:[1,1,2,2,2]" indicates the one component line graph. The five-cycle topology "1:[2,2,2,2,2]" was exceedingly rare, and no conversions for this topology were observed. The conversion scale is the same as for the recruitment figures in the main text. Error bars are 95% confidence intervals. 35

2.7 Recruitment conversion for demographically homogeneous neighborhoods, as a function of (a) 2-node, (b) 3-node, and (c) 4-node contact neighborhood graphs. The conversion scale is the same as for the recruitment figures in the main text. Error bars represent 95% confidence intervals. 36

2.8 Recruitment conversion rates for the nine most frequent pairs of graphs with matched embeddedness distributions. (a) Illustrations of the three most frequent neighborhood graphs pairs with identical embeddedness distributions but differing component counts. The invited node is shown in gray, and the embeddedness distribution refers to the embeddedness of the gray edges. The other six frequent graph topology pairs all contain one of these pairs, up to the addition of a node singleton. (b) The importance of diversity when controlling for embeddedness, examining the nine most common neighborhood graph pairs with identical embeddedness distributions but differing component counts. Each data point is labelled by its degree distribution and its connected component count, as in Figure 2.6. The conversion scale is the same as for the recruitment figures in the main text. Error bars are 95% confidence intervals. 38

2.9 Engagement as a function of diversity in a neighborhood, conditioned on size. For each size $n = 10, 20, 30, 40, 50$, plots are shown that correspond to Figure 2.4(d-f), showing the relative engagement rate as a function of component counts. The right three plots correspond exactly to the plots in Figure 2.4(d-f). All engagement rates are reported on a single relative scale, where 1.0 signifies the average conversion rate across all 50-node neighborhoods. Error bars are 95% confidence intervals. 40

2.10 Controlling for the size of the k-brace. We focus on neighborhoods of size 50 with exactly 35 and 45 nodes in their 1-brace, and again see that engagement is an increasing function of 1-brace component count. All engagement rates are reported on a single relative scale, where 1.0 signifies the average conversion rate across all 50-node neighborhoods. Error bars are 95% confidence intervals. 40

3.1 Subgraph frequencies for three node subgraphs for graphs of size 50, 100, and 200 (left to right). The neighborhoods are orange, groups are green, and events are lavender. The black curves illustrate $G_{n,p}$ as a function of p. 47

3.2 The state transitions diagram for our stochastic graph model with $k = 4$, where γ is the arbitrary edge formation rate, λ is the triadic closure formation rate, and δ is the edge elimination rate. 50

3.3 Subgraph frequencies for 3-node subgraphs in 50-node graphs, shown as a function of p. The black curves illustrate $G_{n,p}$, while the yellow curves illustrate the fit model. 54

3.4 The four-node subgraph frequencies for the means of the 50-node graph collections in Figure 3.3, and the subgraph frequency of the model, fitting the triadic closure rate λ to the mean vectors. As λ increases from $\lambda = 0$ to $\lambda = \lambda_{opt}$, we see how this single additional parameter provides a striking fit. 55

3.5 Subgraph frequencies for 3-node and 4-node subgraphs as function of edge density p. The light green regions denote the asymptotically feasible region found via the linear program. The empirical frequencies are as in Figure 3.3. The black curves illustrate $G_{n,p}$, while the yellow curves illustrate the fit triadic closure model. 62
3.6 Edge densities of neighborhoods, groups, and events as a function of size, n. When $n < 400$, groups are denser then neighborhoods. When $n < 75$, events are denser then neighborhoods. 71

4.1 Sharding the neighbors of a node across N machines. (a) Aggregating properties of the neighbors when the edge list is sharded according to node ID mod N. (b) Aggregating when the edge list is sharded according to a shardmap f. The goal of an efficient shard map is to greatly increase the likelihood that a node is located on the same shard as its neighbors. 79
4.2 Illustration of the constraints for balanced label propagation applied to five shards. Each shard has a two-sided balance constraint, while each pair of shards has a population constraint. 83
4.3 The piecewise-linear utility function for moving nodes from one shard to another, from an example problem iteration. The discontinuous derivates are imperceivable. The utilty approximation shown allows for a significant reduction in the number of constraints in the LP. The red line indicates the threshold found in the optimal solution for balanced propagation: here 8,424 of the 22,728 nodes that wanted to move were moved. 84
4.4 Geometric illustration of the greedy geographic initialization of a shard, with cities as circles with radii indicating cost. The algorithm centers itself at the most costly remaining city, C, and then fills a shard with the cities closest to that city, with a penalty for crossing national borders. When the shard is full, fractional assignments are made. 88
4.5 User traffic differences between countries. Comparing traffic for New Zealand, Italy, and Chile to that of Facebook as a whole, the intraday variability in users accessing the site from a single country far exceeds the variability of the site as whole. 89
4.6 Output of the greedy geographic initialization algorithm for the $\sim 750,000$ known cities, obtaining 234 shards of equal cost, with each shard's most costly city marked. As described in the text, the algorithm is aware of national borders. 90

4.7 Matrices of edges between 78 shards under three different shardings: Balanced label propagation with random initialization, the sharding produced by the geographic initialization, and balanced label propagation after geographic initialization. All matrices share the same logarithmic color scale, saturated to make the structure of the random initialization visible. The fraction of local edges for each shard is also shown. 93
4.8 Iterating the balanced label propagation algorithm with 78 shards, from a geographic and random initialization. Left: the fraction of edges that are local as the balanced propagation is iterated. Right: the fraction of nodes that are moved in each iteration. 93
4.9 Balanced label propagation applied to the LiveJournal social graph, partitioning the graph into 20, 40, 60, 80, and 100 shards. Left: the fraction of edges that are local as the balanced propagation is iterated. Right: the fraction of nodes that are moved in each iteration. 95
4.10 Number of non-local machines queried per request during friend-of-friend calculations in PYMK. The median number of machines queried for the baseline, geographic, and once-propagated shardings were 59, 12, and 9 machines, respectively. 100
4.11 Query time and network traffic for the three different shardings applied to the PYMK service. Because network traffic is instrumented on a machine level, the data also captures daily traffic bursts which correspond to loading data into the service. For this reason, momentary outliers should be considered benign. 100

5.1 Iterating the restreaming partitioning process for static LiveJournal and Orkut graphs. The left column reports results for restreaming LDG, the right column for restreaming FENNEL. Dashed lines are METIS. Iteration zero corresponds to single-shot streaming implementations, though note that FENNEL does not guarantee balance until its final iteration due to ongoing tempering. 125
5.2 The quality of partitions as a function of the number of nodes per shard, for the LiveJournal and Orkut graphs. Notice how the restreamed algorithms essentially match METIS. 127
5.3 The effect of varying α_0 when tempering, where α_0 is the initial value of α and α is increased to the critical α_c over 20 restreams. For $k = 2, 4, 20, 40$ shards, we see that for sufficiently small α_0 the quality of the edge cut does not depend much on the initial α_0 from which the tempering begins. 128
5.4 The tradeoffs between node balance and degree balance when LiveJournal is partitioned into 40 different partitions utilizing several different objectives. We see that stronger notions of balance cost very little in partition quality. 130

5.5 The resulting degree distribution for 4 partitions (colored differently) of LiveJournal from restreaming stratified LDG where portions of the degree distribution are explicitly balanced across 1, 2, 10 and 100 different stratified strata. As the number of strata increases the degree distributions become increasingly similar, though at a small cost in the quality of the edge cut. 130
5.6 The percentage of edges cut for parallelized versions of FENNEL and LDG when partitioning LiveJournal into 40 partitions while parallelized across 2, 10 and 100 machines. Notice that while there is a small cost associated with increasing the number of *machines*, it is small compared to the gains of restreaming. 132

6.1 The probability distribution over the exposure space for a single individual, where the exposure conditions σ_i^0 and σ_i^1 are shown in yellow for both (a) an i.i.d. vertex randomization and (b) an ideal cluster randomization, where the probability mass is collected at exposure conditions of interest. 149
6.2 The cycle graph, (a) where vertices respond $\bar{Y}$ to treatment and 0 to control, shown clustered in groups of $c = 2$ vertices. (b) Asymptotic variance of the estimator for this graph as a function of the number of vertices per cluster, normalized by estimator variance for $c = 1$ vertices per cluster. (c) Simulated variance of the estimator for kth powers of the cycle graph for $k = 1, \ldots, 5$ as a function of the number of vertices per cluster. For each k the variance for cluster size $c = 2k + 1$ grows linearly in k. 153

CHAPTER 1

INTRODUCTION

"Alas! Forgetful of a husband's home duties I again became involved in the dissipated social network, whose fatal meshes too surely entangled me, and unfitted me for that active exertion which was now rendered doubly necessary."
– John Bartholomew Gough, *Autobiography* (1846)

The quantitative study of the structure of social systems dates back to at least the 1930's, when the social scientist Jacob Moreno introduced the *sociogram*, a graphical representation of ties between individuals to illustrate the structure of social groups, more modernly referred to as a *social network* [113]. While the notion that relationships in society form something akin to a "network" is a metaphor that significantly predates Moreno — see the epigraph above — his initial work on sociograms in his 1934 book *Who Shall Survive?* introduced much more than just a metaphor or visual tool, and in fact developed precursors to many of the modern tools used for analyzing social networks today.

Moreno's work was inspired by investigations he conducted as Director of Research at the New York Training School for Girls reformatory school in Hudson, NY, where he sought to broadly study determinants of human behavior. Moreno wondered if there were structural explanations for why certain young girls were running away from the school, and thought that sociographic analysis might hold an answer. He was a careful thinker, and beyond merely analyzing the networks he observed, he was quick to seek notions of statistical significance in his analyses. As part of his earliest work, he introduced what he called *chance sociograms* and what we would now call *random graphs*, used in permutation tests to see if the structures he was observing were statistically significant. Analyses performed against such randomized baselines are now the cornerstone of many modern measures on social networks, such as network modularity [119], and many of the computational tools for analyzing modern large-scale social net-

works rely on appropriate constructed random graphs, e.g. random graphs with arbitrary degree distributions [117].

Concurrent with Moreno's development of sociometry and the 1934 publication of *Who Shall Survive?*, in 1936 the Hungarian mathematician Dénes Kőnig published the first organized textbook on the emerging mathematical field of graph theory, *Theorie der endlichen und unendlichen Graphen*. The study of graphs is generally considered to have begun in 1736 with Leonhard Euler's analysis of the "Seven Bridges of Königsberg problem" [48, 71], and has a rich history that predates Kőnig's textbook [21], but it was Kőnig's textbook that first organized the discipline. It is fortuitous that this should happen concurrently with Moreno's initial investigations.

As the study of sociograms and sociometry began to take off in the 1950's, mathematicians such as Anatol Rapoport and Frank Harary successfully led efforts to apply the mathematical language of graph theory to social networks [128, 67, 30, 69]. At the end of the 1950's, the rigorous study of random graphs initiated by Paul Erdős and Alfred Rényi [47] laid further foundations for studying social networks as mathematical objects. By studying probability distributions over the space of graphs, random graphs made it possible to view Moreno's work on "chance sociograms" in a rigorous light.

Since these early syntheses, the study of social networks has benefited from a symbiotic relationship with graph theory, which has contributed important operational tools for analyzing social theories. A notable early example of such *operationalization* — providing precise definitions and measures that enable quantitative analysis — was the work by James Davis and Samuel Leinhardt published in 1967. In that work, Davis and Leinhardt set out to operationalize a theory of small social groups put forward in 1950 by George Homans, contained in his book *The Human Group* [68]. Homans' theory proposed that small groups of people inevitably generate a social structure that contains many clique subgroups and a ranking system. Through their work, Davis and Lein-

hardt provided an "operational statement of Homans' theory", that "seven key triads are less frequent than the random model would predict" [44], the infrequent triads being those triads that a graph would be free from if it consisted only of clique subgraphs and directed relationships aligned with an hierarchy.

Davis and Leinhardt's work was importantly exceptional in that it employed an "electronic computer" to perform the involved analysis — in 1967(!) — of adjacency matrices representing 427 groups, and 60 simulated groups with random relationships. While the mathematics of random graphs provided a language for describing Homans' theory, it was the computational perspective gained through data analysis that was the true essence of their landmark operationalization. Recently, Homans' social theory of cliques and ranking was re-examined within the context of the National Longitudinal Study of Adolescent Health [131], examining over 90,000 students across 84 school networks in a manner similar to Davis and Leinhardt. This recent work corroborated the findings of Davis and Leinhardt, while also introducing a maximum likelihood inference framework for inferring rankings within the theory [16], a computationally intensive line of inquiry far beyond what Davis and Leinhardt could have hoped to achieve in 1967.

The early computational analyses of Davis and Leinhardt recognized that social networks are complicated mathematical objects, and any manual analysis was intractable even for moderately sized networks. In the decades the followed, the field of social network analysis developed a rich set of computational tools to analyze various social theories computationally [158], but for the most part these analyses did not attract much attention from mathematicians or computer scientists.

As the 20th century drew to close, however, a broad range of other research communities turned their attention to diverse research problems centered around large-scale graphs. In the narrow window between 1997 and 2000, significant advances were made by mathematicians studying the theory of random graphs and percolation theory

[111, 5], applied mathematicians studying synchronization of oscillators [159], statistical physicists studying scaling laws [141, 17, 6, 73], and computer scientists from subfields including artificial intelligence, information retrieval, and data mining all studying various combinations of web mining [42, 60], link analysis [86, 123], graph partitioning [79], and communication network architectures [49].

Some of these early analyses did in fact extend to social networks, notably the Watts-Strogatz model of "small-world networks" that introduced a class of sparse graphs concurrently exhibiting both a small diameter and clustering (many triangles). Before small-world networks were introduced, it was known that sparse Erdős-Rényi random graphs exhibited a small diameter with high probability and that lattice graphs exhibited clustering, while empirically it was known that social networks exhibited both these properties, but there was no good model of sparse graphs exhibiting the two traits concurrently. By introducing the *clustering coefficient* of a graph as a definition of clustering, and by showing that small-world networks exhibit a high clustering coefficient and a small diameter, Watts and Strogatz contributed an important measure of how social network models can be deemed realistic.

In addition to analyzing their theoretical model of small-world networks, Watts and Strogatz analyzed three datasets: a network of film actors collaborations, the electrical power grid of the western United States, and the neuronal network of the nematode worm *C. elegans*. The network of film actors was quite large, documenting collaborations between more than 225,000 actors using data available from the *Internet Movie Database* (IMDB) as of April 1997. While the traditional social network analysis community had been studying small-scale network datasets for decades, the work by Watts and Strogatz coincided with the availability of large-scale datasets being organized on the world wide web, including the link structure of the web itself [6, 73] and the network structure of the internet itself [49, 5]. While the early web was principally orga-

nized around documents and information, it soon became clear that large portions of the internet were in fact encoding human relationships and other human behavior. In the presence of data, the scale of social networks being analyzed was perched to grow by orders of magnitude.

Research efforts gradually began targeting increasingly ambitious scales of social network structure, a progression that in turn required increasing computational sophistication. Some efforts focussed on the social network of academic collaborators, studying first smaller networks of researchers affiliated with a single institution [61], and in 2003 Liben-Nowell and Kleinberg used a dataset of 23,500 authors and 100,000 edges to frame the problem of link prediction in social networks [99]. Another source of social relations that quickly landed under the microscope was e-mail: in 2004 a corpus of over 600,000 emails between 158 employees of the Enron Corporation was released to the public through the Federal Energy Regulatory Commission, and subsequently became an object of frequent study [85, 107], while another major study by Kossinets and Watts examined the email exchanges of 43,000 students, faculty, and staff at a large university [89]. As more and more individuals set up personal homepages on the web, in 2003 it occured to Adamic and Adar to study the links between 9,700 homepages at Stanford and at MIT as social networks [1]. The link structure of personal weblogs was next, and it was here that scales first became truly overwhelming. In 2005, Adamic and Glance studied 1,494 political blogs in the wake of the 2004 election [2], while later the same year the analysis of 1.03 million weblogs indexed through blogdex was released [106], and two separate studies of over 1 million blogs hosted at LiveJournal were used to perform detailed analyses of geographic assortativity [100] and of the social process of group formation [10]. To both gather and analyze graph data of these scales, computational naiveté was no longer admissible.

In a few short years, computational perspectives had made it possible to study social

networks at tremendous scales. In many ways the 21st century's first decade offered a golden age of large-scale observational data analysis: because no one had ever assembled datasets of these magnitudes before, answering basic questions in an observational setting made it possible to propose answers, albeit coarse answers, where no previous attempt at answers had been possible. In 2006, basic questions of collective action and group dynamics studied in the 1970's by Granovetter [64] and Schelling [134] were given computational reformulations and analyzed at scale in the context of LiveJournal group formation by Backstrom and colleagues [10]. In 2008, Stanley Milgram's creative investigations of the 1960's into the six degrees of separation phenomenon by initiating 160 chain letters [108] was reexamined at a planetary scale through the MSN online chat network [95] (and more recently on Facebook [14]). As the study of large-scale social networks progressed to encompass not just small groups or samples but various notions of "all the data," for a while it nearly seemed as though any social theory could be tested by studying large enough observational social network datasets.

Unfortunately, there are limits to what an observational data analysis can discern, no matter how large the dataset. While more and more data makes it possible to provide increasingly accurate predictive understandings of nearly any question related to that data, social theories often concern themselves with causal/explanatory relationships, not merely predictive/descriptive relationships. Most particularly, theories of *social influence* are specifically theories about a causal relationship. In line with these concerns, the more recent era of analyses examining large-scale social networks has been attempting to target causal understandings, when possible.

In 2007, Christakis and Fowler released a large-scale analysis of social contagion within the 12,000 study participants in the longitudinal Framingham Heart Study, claiming that obesity was socially contagious [38]. Christakis and Fowler were interested in the role of social influence in public health, and realized that the Framingham study,

which had been running for over thirty years, had been surreptitiously collecting social relationship information as part of its periodic surveys over these thirty years, providing a convenient chance to test various theories of how individuals influence each others' health through social relationships. The difficulty with understanding social influence in social settings is that, even if they could have gone back in time to design the study to include a randomized trial, or even if they were able to wait 30 years for a new study to be conducted, it is not possible to reliably randomize individuals' social ties. Recognizing that a randomized trial would be difficult, they hoped that by applying a thoughtful methodology they could tease out a causal understanding of how the health of individuals influence each other through time. Unfortunately, a careful analysis by Shalizi and Thomas in 2011 showed that, as the title of that work clearly communicated, "homophily and contagion are generically confounded in observational social network studies" [135]. Shalizi and Thomas showed that it was not possible to distinguish homophily — the notion that like attracts like and that similar people are more likely to be friends — from social influence. Christakis and Fowler had established that being friends with an obese person predicted that you would become obese, but attempts to establish causality inherently ran up against the obstacles discovered by Shalizi and Thomas.

The research conversations surrounding the Christakis and Fowler study, including the Shalizi and Thomas negative result, rightly gave the large-scale social networks research community pause for thought. The conversation initiated a rapid shift towards more careful experimental approaches that married large-scale analysis with causal inference. As another analogy, in 2007 Centola and Macy had initiated a successful research program studying complex contagion, the notion in collective action that individuals do not respond to social cues independently or linearly, but they perhaps respond superlinearly as a function of the number of peers providing cues. Through simula-

tion analyses [32, 33], they had shown that complex contagion behaved very differently than simple contagion on various network topologies. In 2010 and 2011, Centola published two randomized experiments to establish that human subjects did in fact make decisions consistent with complex contagion [34, 35]. Other important work, including by Aral and colleagues [8], introduced the social networks community to propensity score matching, an established methodology developed in the 1980's by the statistical inference community [132] for reducing bias in observational studies.

In recent years, the widespread application of randomized experiments to large-scale social networks studies has been tremendously successful. A large-scale hold-out experiment conducted in 2010 on Facebook was able to quantify the significant role of homophily in information diffusion [15], the general confounder that Shalizi and Thomas had identified in observational studies of influence. The important role of social factors in voter turn-out was similarly examined at scale through Facebook in a 61-million participant experiment led by Fowler and colleagues, this time employing a randomized experiment [23]. Throughout these advances, computational perspectives have been at the heart of operationalizing diverse theories of social and human behavior, and as studies target increasingly sophisticated hypotheses, including those pertaining to causal relationships, the demand for computational machinery only increases.

It is important to emphasize that the negative result of Shalizi and Thomas, that homophily and social influence are generally confounded, is specifically a critique of studies of social influence. It is certainly not a blanket dismissal of non-causal investigations of social networks. First, there are many social theories that are not causal, such as basic structural claims of connectivity, clustering, and assortativity. Second, there are many questions where causal experiments are not possible, such as randomizing who people are friends with. The questions posed by Christakis and Fowler are still of grave consequence for public health, and the best possible answers, with known caveats, are

certainly of great interest.

Beyond providing a quantitative language for existing social theories, computational perspectives also contribute their own findings. The investigations of network ranking and centrality that helped transform search engines such as Google into the backbones of the modern web [123] have also had exciting implications for understanding social networks. When tracking infectious diseases, it has been postulated [41] and shown [39] that rather than tracking random individuals, there are significant advantages to tracking friends of random individuals, a strategy that explicitly seeks out nodes of a social network with higher centrality. The use of such "friend sensing" has been shown empirically to track a flu epidemic on a college campus with a two week lead time compared to tracking random individuals [39]. In another vein, link prediction algorithms [99] have been tremendously useful to online social networking services such as Facebook and LinkedIn. In the context of link prediction, the fundamental question is precisely predictability, and not causality, and Facebook has employed various observational findings regarding assortative mixing in geographic [100] and other latent spaces to improve friend recommendations through their "People You May Know" service. These recommendations ultimately drives 40% of all friendships formed on Facebook. Overall, the study of social networks has come to increasingly rely on computational perspectives, and it has been the goal of this thesis to push the means of analysis forward for a new generation of both insights and applications. The contributions in this thesis are variously distributed, and were developed to enrich several of the points in the above overview of the field.

1.1 Contributions

Chapter 2 presents work in the tradition of operationalizing a social theory through a computational perspective, examining the role of diversity in social decision-making by

studying the growth of the Facebook online social networking service. The growth of Facebook as a service constitutes a rare social contagion process with genuinely global adoption, now with over one billion active users. In 2010 I initiated a collaboration with Facebook to study the role of graph structure in this rapid growth. Traditional models of social contagion have been based on analogies with biological contagion, where the probability that an individual is affected by the contagion grows monotonically with the number of affected individuals with whom he or she is in contact. The role of graph structure — how the affected neighbors are related — has been challenging to evaluate due to the difficulty in obtaining detailed data on individual network neighborhoods during the course of a large-scale contagion process. We found that the spread of Facebook departed notably from traditional epidemiological models of contagion. In particular, we observed that the probability of contagion was tightly controlled by the number of connected components in an individual's contact neighborhood — the "structural diversity" — rather than by the actual size of the neighborhood. We also examined the role of structural diversity in long-term user engagement: studying the network neighborhoods that new users formed after one week, we found that users with many distinct and substantive components in their network neighborhood were much more likely to become engaged users, logging in at least six out of seven days per week. Similar to the health questions examined by Christakis and Fowler, randomizing the structure of people's friendships was not an admissible strategy.

The work presented in Chapter 3 is closely related to the work of Davis and Leinhardt, who used subgraph frequencies compared against a random graph baseline as an operational measure of Homans' theory of group formation. The work in Chapter 3 shows how, in the study of dense social networks such as groups or network neighborhoods, combinatorial structure constrains graph statistics in significant and non-trivial ways. The key question driving that work was to ask: when studying social networks,

what properties are "social" properties and what are "network" properties? Using the theory of graph homomorphisms, we studied the combinatorial constraints on frequencies of induced subgraphs, that statistics studied by Davis and Leinhardt and in many other contexts since. We found that the space of subgraph frequencies is governed both by its combinatorial properties, based on extremal results that constrain all graphs, as well as by its empirical properties of *social* graphs, manifested in the way that real social graphs appear to lie near a simple one-dimensional "human manifold" through this space. In particular, we show that the frequency of the subgraph sometimes referred to as the "forbidden triad" — three people with two social relationships between them but one absent relationship — has a non-trivial upper bound in not just social graphs, but in all graphs: no sufficiently large graph can possess forbidden triads at a frequency greater than $3/4 + o(1)$ of all triads. More generally, we showed that any k-node subgraph that is not a complete or empty subgraph has a frequency among k-node subgraphs that is bounded away from one. Thus, there is an extent to which almost all subgraphs are mathematically "forbidden" from occurring beyond a certain frequency. For dense graph contexts such the study of network neighborhoods or social groups, these combinatorial constraints that govern subgraph frequencies provide a significant new perspective for understanding when observed structures indicate social structure and not merely structures that constrain all graphs.

In Chapters 4 and 5, we ask: can the abundance of modern insights about social networks be applied to improve the efficiency of social data analysis? How can we "close the loop" on the study of social data, using insights to make analysis more efficient? The work in Chapter 4, conducted in collaboration with Facebook, developed a way to use social graph structure to dramatically improve the partitioning of the distributed computing infrastructure supporting Facebook's "People You May Know" machine learning system. By using a novel graph partitioning algorithm that we developed, which we call

"balanced label propagation," we were able to harness the locally clustered structure of social graphs while still achieving a global balancing objective. Graph partitioning has a very rich literature, but existing algorithms typically cannot scale to billions of edges, or cannot provide guarantees about partition sizes. This algorithm combined the computational efficiency of local label propagation — where nodes are iteratively relabeled to the same "label" as the plurality of their graph neighbors — with the guarantees of constrained optimization, guiding the propagation by a linear program constraining the partition sizes. By initializing this local algorithm using a geographic partitioning of the graph, we were ultimately able to achieve very strong performance gains: in a system distributed across 78 machines, the median number of machines queried per request fell from 60 to 9, when compared to the previous naive random sharding. The average query times and average network traffic levels were reduced by 50.5% and 37.1%, respectively.

The work presented in Chapter 5, a separate collaboration not involving Facebook, shows how light-weight streaming partitioning algorithms making local decisions can efficiently divide graphs into balanced disjoint partitions while also satisfying balancing objectives much more sophisticated than node count. The partitioning not only provides a good graph cut, but it can scalably do so while also balancing node attributes, such as gender, age, or degree. As an example, we apply the technique to partitioning graphs such that all the subsets have the same degree distribution. Being a local algorithm that harnesses local structure, the relative performance of this streaming algorithm was again strongest for social graph data.

The work presented in Chapter 6 engages directly with the challenges of causal inference on networks described in this introduction, developing a new framework for designing and analyzing large-scale network experiments, in situations where randomized experiments are feasible. Specifically, the standard goal of experiments is to causally estimate the "treatment effect" of a node-level intervention by testing it on a random

sample of the overall population. Inference in these settings becomes much more difficult when the experimental treatment of individuals spills over to neighboring individuals along an underlying social network. In the absence of the so-called "stable unit treatment value assumption" (SUTVA), accurate inference requires new modeling assumptions about social network structure and/or social behavior. The framework that this work proposes for network experimentation relies heavily on graph partitioning. As a result, we show how the approaches developed in Chapters 4 and 5, and in fact any previous work on graph partitioning, can be tremendously important to designing low variance experiments on networks.

The work begins by characterizing graph-theoretic conditions under which individuals can be considered to be "network exposed" to an experiment, operationalizing assumptions about social behavior. Given these assumptions as part of our analysis strategy, the work then focused on designing graph clustering algorithms with favorable estimator variance properties. For any graph clustering, our strategy of *graph cluster randomization* admits an efficient exact dynamic program for computing the probabilities for each node being network exposed under our exposure conditions. These exact probabilities therefore facilitate an unbiased effect estimator (provided that the exposure model has been properly specified) by using inverse-probability weights. Under a restricted-growth assumption on the growth rate of graph neighborhoods, we showed that a simple constructive clustering algorithm based on node neighborhoods provides an estimator with a variance that can be upper bounded by a linear function of the graph degrees. We do not provide any variance guarantees for the algorithms given in Chapter 4 and 5, but in practice they may be much more performative. In particular, the ability of the algorithm in Chapter 5 to balance node attributes through stratification provides a promising variance reduction strategy. Establishing reasonable assumptions about social structure and behavior will be key to providing both rigorous experimentation tools

for social science questions and for many other new computing applications.

CHAPTER 2

STRUCTURAL DIVERSITY IN SOCIAL CONTAGION

The concept of contagion has steadily expanded from its original grounding in epidemic disease to describe a vast array of processes that spread across networks, notably social phenomena such as fads, political opinions, the adoption of new technologies, and financial decisions. Traditional models of social contagion have been based on physical analogies with biological contagion, in which the probability that an individual is affected by the contagion grows monotonically with the size of his or her "contact neighborhood" — the number of affected individuals with whom he or she is in contact. Whereas this contact neighborhood hypothesis has formed the underpinning of essentially all current models, it has been challenging to evaluate it due to the difficulty in obtaining detailed data on individual network neighborhoods during the course of a large-scale contagion process. Here we study this question by analyzing the growth of Facebook, a rare example of a social process with genuinely global adoption. We find that the probability of contagion is tightly controlled by the number of connected components in an individual's contact neighborhood, rather than by the actual size of the neighborhood. Surprisingly, once this "structural diversity" is controlled for, the size of the contact neighborhood is in fact generally a negative predictor of contagion. More broadly, our analysis shows how data at the size and resolution of the Facebook network make possible the identification of subtle structural signals that go undetected at smaller scales yet hold pivotal predictive roles for the outcomes of social processes.

2.1 Introduction

Social networks play host to a wide range of important social and nonsocial contagion processes [125, 118, 45, 10, 82, 160, 38, 145]. The microfoundations of social contagion can, however, be significantly more complex, as social decisions can depend much more

subtly on social network structure [134, 64, 27, 89, 32, 33, 124, 8, 57]. In this study we show how the details of the network neighborhood structure can play a significant role in empirically predicting the decisions of individuals.

We perform our analysis on two social contagion processes that take place on the social networking site Facebook: the process whereby users join the site in response to an invitation e-mail from an existing Facebook user (henceforth termed "recruitment") and the process whereby users eventually become engaged users after joining (henceforth termed "engagement"). Although the two processes we study formally pertain to Facebook, their details differ considerably; the consistency of our results across these differing processes, as well as across different national populations (see Section 2.5), suggests that the phenomena we observe are not specific to any one modality or locale.

The social network neighborhoods of individuals commonly consist of several significant and well-separated clusters, reflecting distinct social contexts within an individual's life or life history [138, 63, 28]. We find that this multiplicity of social contexts, which we term structural diversity, plays a key role in predicting the decisions of individuals that underlie the social contagion processes we study.

We develop means of quantifying such structural diversity for network neighborhoods, broadly applicable at many different scales. The recruitment process we study primarily features small neighborhoods, but the on-site neighborhoods that we study in the context of engagement can be considerably larger. For small neighborhoods, structural diversity is succinctly measured by the number of connected components of the neighborhood. For larger neighborhoods, however, merely counting connected components fails to distinguish how substantial the components are in their size and connectivity. To determine whether the structural diversity of on-site neighborhoods is a strong predictor of on-site engagement, we evaluate several variations of the connected component concept that identify and enumerate substantial structural contexts within large

neighborhood graphs. We find that all of the different structural diversity measures we consider robustly predict engagement. For both recruitment and engagement, structural diversity emerges as an important predictor for the study of social contagion processes.

2.2 User recruitment

To study the spread of Facebook as it recruits new members, we require information not just about Facebook's users but also about individuals who are not yet users. Thus, suppose that an individual A is not a user of Facebook; it is still possible to identify a set of Facebook users that A may know because these users have all imported A's e-mail address into Facebook. We define this set of Facebook users possessing A's e-mail address to be A's contact neighborhood in Facebook. This contact neighborhood is the subset of potential future friendship ties that can be determined from the presence of A's e-mail address (Figure 2.1a). Whereas A may in fact know many other people on Facebook as well, such additional friendship ties remain unknown for individuals who do not choose to register and so cannot be studied as a predictor of recruitment. The e-mail contact neighborhoods we study are generally quite small, typically on the order of five or fewer nodes.

We can now study an individual's decision to join Facebook as follows. Facebook provides a tool through which its users can e-mail friends not on Facebook to invite them to join; such an e-mail invitation contains not only a presentation of Facebook and a profile of the inviter, but also a list of the other members of the individual's contact neighborhood. We analyze a corpus of 54 million such invitation e-mails, and the fundamental question we consider is the following: How does an individual's probability of accepting an invitation depend on the structure of his or her contact neighborhood?

Traditional hypotheses suggest that this probability should grow monotonically in the size of the contact neighborhood [45, 134, 64]. What we find instead, however, is

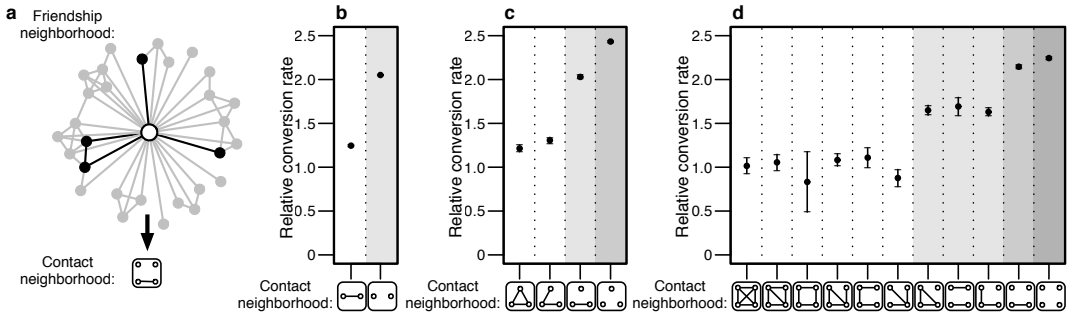

Figure 2.1: Contact neighborhoods during recruitment. (a) An illustration of a small friendship neighborhood and a highlighted contact neighborhood consisting of four nodes and three components. (b–d) The relative conversion rates for two-node, three-node, and four-node contact neighborhood graphs. Shading indicates differences in component count. For five-node neighborhoods, see Figure 2.6. Invitation conversion rates are reported on a relative scale, where 1.0 signifies the conversion rate of one-node neighborhoods. Error bars represent 95% confidence intervals and implicitly reveal the relative frequency of the different topologies.

a striking stratification of acceptance probabilities by the number of connected components in the contact neighborhood (Figure 2.1b–d and Figure 2.6). When going beyond component count, one may suspect that edge density has a significant impact on the recruitment conversion rate: Among the single-component neighborhoods of a given size, there is a considerable structural difference between neighborhoods connected as a tree and those connected as a clique. However, within the controlled conditional datasets of one-component neighborhoods of sizes 4–6, we see that edge density has no discernible effect (Figure 2.2a).

Moreover, we see that once component count is controlled for (Figure 2.2b), neighborhood size is largely a negative indicator of conversion. In effect, it is not the number of people who have invited you, nor the number of links among them, but instead the number of connected components they form that captures your probability of accepting the invitation. Note that this analysis has been performed in aggregate and thus unavoidably reflects the decisions of different individuals. The ability to reliably estimate acceptance probabilities as a function of something as specific as the precise topology

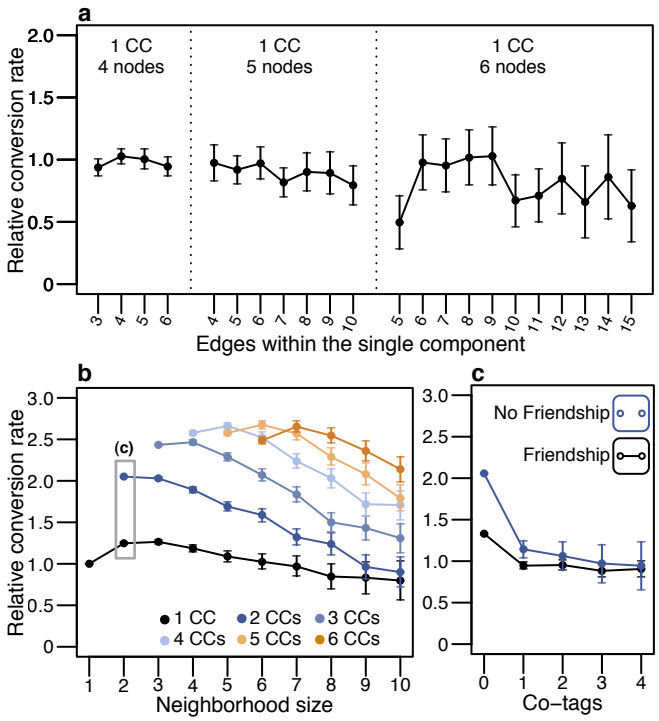

Figure 2.2: Recruitment contact neighborhoods and component structure. (a) Conversion as a function of edge count neighborhoods with one connected component (1 CC) with four to six nodes, where variations in edge count predict no meaningful difference in conversion. (b) Conversion as a function of neighborhood size, separated by CC count. When component count is controlled for, size is a negative indicator of conversion. (c) Conversion as a function of tie strength in two-node neighborhoods, measured by photo co-tags, a negative indicator of predicted conversion. Recruitment conversion rates are reported on a relative scale, where 1.0 signifies the conversion rate of one-node neighborhoods. Error bars represent 95% confidence intervals.

of the contact neighborhood is possible only because the scale of the dataset provides us with sufficiently many instances of each possible contact neighborhood topology (up through size 5).

We view the component count as a measure of "structural diversity," because each connected component of an individual's contact neighborhood hints at a potentially distinct social context in that individual's life. Under this view, it is the number of distinct social contexts represented on Facebook that predicts the probability of joining. We show that the effect of this structural diversity persists even when other factors are

controlled for. In particular, the number of connected components in the contact neighborhood remains a predictor of invitation acceptance even when restricted to individuals whose neighborhoods are demographically homogeneous (in terms of sex, age, and nationality; Figure 2.7), thus controlling for a type of demographic diversity that is potentially distinct from structural diversity. The component count also remains a predictor of acceptance even when we compare neighborhoods that exhibit precisely the same mixture of "bridging" and "embedded" links (Figure 2.8), the key distinction in sociological arguments based on information novelty [63, 28].

For contact neighborhoods consisting of two nodes, we observe that the probability an invitation is accepted is much higher when the two nodes in the neighborhood are not connected by a link (hence forming two connected components, Figure 2.1b) compared with when they are connected (forming one component). Is there a way to identify cases where people are likely to know each other, even if they are not linked on Facebook? The photo tagging feature on Facebook suggests such a mechanism. Photographs uploaded to Facebook are commonly annotated by users with ?tags? denoting the people present in the photographs. We can use these tags to deduce whether two unlinked nodes in a contact neighborhood have been jointly tagged in any photos, a property we refer to as ?co-tagging,? which serves as an indication of a social tie through copresence at an event [43].

Using photo co-tagging, we find strong effects even in cases where the presence of a friendship tie is only implicit. If a contact neighborhood consists of two unlinked nodes that have nevertheless been co-tagged in a photo, then the invitation acceptance probability drops to approximately what it is for a neighborhood of two linked nodes (Figure 2.2c). In other words, being co-tagged in a photo indicates roughly the same lack of diversity as being connected by a friendship link. We interpret this result as further evidence that diverse endorsement is key to predicting recruitment. Meanwhile,

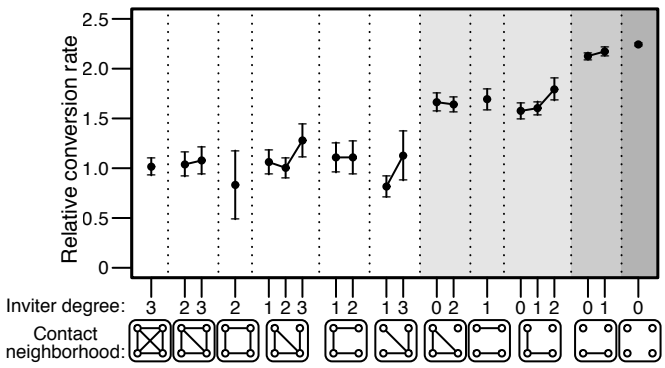

Figure 2.3: Inviter position during recruitment. Shown is recruitment conversion as a function of neighborhood graph topology and inviter position in neighborhoods of size 4. The position of the inviter within the neighborhood graph is described exactly (up to symmetries) by node degree. Shading indicates differences in component count. Recruitment conversion rates are reported on a relative scale, where 1.0 signifies the conversion rate of one-node neighborhoods. Error bars represent 95% confidence intervals.

when the two nodes are friends, co-tags offer a proxy for tie strength, and we see that if the two nodes have also been co-tagged, then the probability of an accepted invitation decreases further. From this we can interpret tie strength as an extension of context, because two strongly tied nodes plausibly constitute an even less diverse endorsement neighborhood.

Finally, we study the position of the inviter within the neighborhood topologies. When studying recruitment, one might suspect that the structural position of the inviter — the person who extended the invitation — might signify differences in tie strength with the invitee and therefore might significantly affect the predicted conversion rate. We find that inviter position figures only slightly in the conversion rate (Figure 2.3), with invitations stemming from a high-degree position in the contact neighborhood predicting only a slightly higher conversion rate than if the inviter is a peripheral node.

2.3 User engagement

Participation in a social system such as Facebook is built upon a spectrum of social decisions, beginning with the decision to join (recruitment) and continuing on to decisions about how to choose a level of engagement. We now show how structural diversity also plays an analogous role in this latter type of decision process, studying long-term user engagement in the Facebook service. Whereas recruitment is a function of the complex interplay between multiple acts of endorsement, engagement is a function of the social utility a user derives from the service. Our study of engagement focuses on users who registered for Facebook during 2010, analyzing the diversity of their social neighborhoods 1 week after registration as a basis for predicting whether they will become highly engaged users 3 months later. Users are considered engaged at a given time point if they have interacted with the service during at least 6 of the last 7 days. Facebook had 845 million monthly active users on December 31, 2011, and during the month of December 2011, an average of 360 million users were active on at least 6 out of the last 7 days. We define engagement on a weekly timescale to stabilize the considerable weekly variability of user visits. Our goal is therefore to predict whether a newly registered user will visit Facebook at least 6 of 7 days per week 3 months after registration.

Friendship neighborhoods on Facebook are significantly larger than the e-mail contact neighborhoods from our recruitment study. We focus our engagement study on a population of $\sim$10 million users who registered during 2010 and had assembled neighborhoods consisting of exactly 10, 20, 30, 40, or 50 friends 1 week after registration. For social network neighborhoods of this size, we find that a neighborhood containing a large number of connected components primarily indicates a large number of one-node components, or "singletons", and as such, it is not an accurate reflection of social context diversity.

To address this, we evaluate three distinct parametric generalizations of component

count. First, we measure diversity simply by considering only components over a certain size k. Second, we measure diversity by the component count of the k-core of the neighborhood graph [22], the subgraph formed by repeatedly deleting all vertices of degree less than Graphic. Third, we define a measure that isolates dense social contexts by removing edges according to their *embeddedness*, the number of common neighbors shared by their two endpoints; intuitively this is an analog, for edges, of the type of node removal that defines the k-core. Adapting earlier work on embeddedness by Cohen [40], we define the k-brace of a graph to be the subgraph formed by repeatedly deleting all edges of embeddedness less than Graphic and then deleting all single-node connected components. (Cohen's work was concerned with a definition equivalent to the largest connected component of the k-brace; because we deal with the full subgraph of all nontrivial components, it is useful to adapt the definitions as needed.) Examples of these three measures applied to a neighborhood graph are shown in Figure 2.4a and 2.4b, illustrating the connected components of size 3 or greater, the connected components of the 2-core, and the connected components of the 1-brace. We see that the three parametric measures we evaluate differ measurably in how they isolate "substantial" social contexts.

The k-core component count for Graphic is simply the component count of the original graph, the same as we analyzed when examining recruitment. For $k = 1$, the k-core component count is the count of nonsingleton components, whereas for $k = 2$, all tree-like components are discarded and the remaining components are counted. When considering the k-brace, observe that for all graphs the k-brace is a subgraph of the $(k + 1)$-core: indeed, because each node in the k-brace is incident to at least one edge, and each edge in the k-brace has embeddedness at least $k + 1$, all nodes in the k-brace must have degree at least $k + 1$. It is therefore reasonable to compare the 1-brace to the 2-core. Both of these restrictions discard tree-like components, but the 1-brace will

tend to break up components further than the 2-core does — the operation defining the 1-brace continues to cleave components in cases where sets of nodes forming triangles are linked together by unembedded edges or where a component contains cycles but no triangles. The notion of the k-core has been applied both to the study of critical phenomena in random graphs [104, 75] and to models of the Internet [7, 29], but to our knowledge the k-brace has not been studied extensively (see Section 2.6 for some basic results on the k-brace and [40] for analysis of a related definition).

When studying the structural diversity of 1-week Facebook friendship neighborhoods as a predictor of long-term engagement, simply counting connected components leads to a muddled view of predicted engagement (Figure 2.4c). However, extending the notion of diversity according to any of the definitions above suffices to provide positive predictors of future long-term engagement. Specifically, when considering the components of the 1-brace, which removes small components and severs unembedded edges, we see that diversity (captured by the presence of multiple components) emerges as a significant positive predictor of future long-term engagement (Figure 2.4f). We also see that the closely related 2-core component count is a clean predictor (Figure 2.4e). Finally, if we consider simply the number of components of size k or larger in the original neighborhood (without applying the core or brace definitions), we see that small values of k are not enough (Figure 2.4d); but even here, when k is increased to make the selection over components sufficiently astringent (in particular, when we count only components of size 8 or larger), a clean indicator of engagement again emerges.

When considering the k-brace, it is sufficient to consider the component count of the 1-brace for our purposes, but larger values of k may be useful for analyzing larger neighborhoods in other domains. We note that the presence of several components in the k-core and the k-brace is fundamentally limited by the size of the core/brace, and we perform a control of this potentially confounding factor (Figure 2.10). The conventional

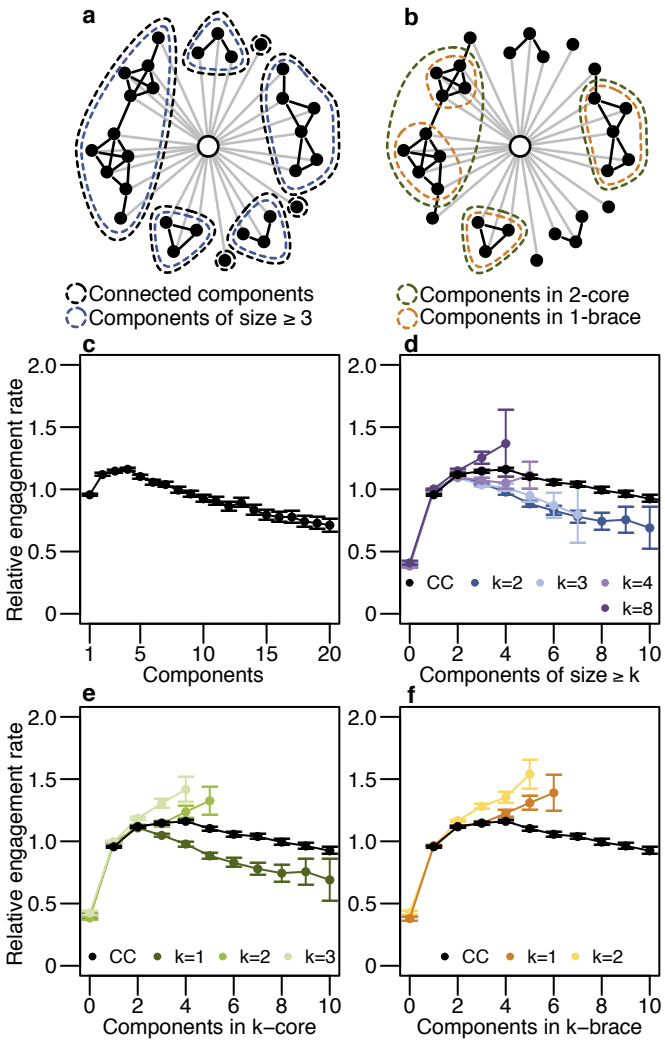

Figure 2.4: Engagement and structural diversity for 50-node friendship neighborhoods. (a) Illustration of the connected components in a friendship neighborhood, delineating connected components and components of size ≥ 3. (b) Illustration of the k-core and the k-brace, delineating the connected components of the 2-core and the 1-brace. (c) Engagement as a function of connected component count. (d) Engagement as a function of the number of components of size $\geq k$, for $k = 2, 3, 4, 8$, with connected component (CC) count shown for comparison. (e) Engagement as a function of k-core component count for $k = 1, 2, 3$, with CC count shown for comparison. (f) Engagement as a function of k-brace component count for $k = 1, 2$, with CC count shown for comparison. Engagement rates are reported on a relative scale, where 1.0 signifies the average conversion rate of all 50-node neighborhoods. All error bars are 95% confidence intervals. For other neighborhood sizes, see Figure 2.9.

wisdom for social systems such as Facebook is that their utility depends crucially upon the presence of a strong social context. Our findings validate this view, observing that

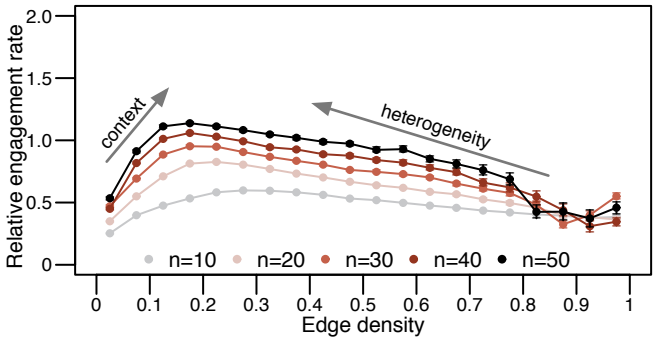

Figure 2.5: Engagement as a function of edge density. For five different neighborhood sizes, $n = 10, 20, 30, 40, 50$, we see that when component count is not accounted for, an internal engagement optimum is observed, showing the combined forces of focused context and structural heterogeneity. Engagement rates are reported on a relative scale, where 1.0 signifies the average conversion rate of all 50-node neighborhoods. All error bars are 95% confidence intervals.

the predicted engagement for users who lack any strong context (e.g., those who have zero components in their neighborhood 1-brace) is much lower than for those with such a context. Our analysis importantly extends this view, finding that the presence of multiple contexts introduces a sizable additional increase in predicted engagement.

A cruder approach to diversity might consider measuring diversity through the edge density of a neighborhood, figuring that sparse neighborhoods would be more varied in context. In Figure 2.5 we see how this approach results in a complicated view where the optimal edge density for predicting engagement lies at an internal and size-dependent optimum. Given what our component analysis reveals, we interpret this observation as a superposition of two effects: Too few edges imply a lack of context [10] but too many edges imply a lacking diversity of contexts, with a nontrivial interior clearly dominating the boundary conditions. From Figure 2.5 it also becomes clear that internal neighborhood structure is at least as important as size, with a 20-node neighborhood featuring a well-balanced density predicting higher conversion than a sparse or dense 50-node neighborhood.

2.4 Discussion

Detailed traces of Facebook adoption provide natural sources of data for studying social contagion processes. Our analysis provides a high-resolution view of a massive social contagion process as it unfolded over time and suggests a rethinking of the underlying mechanics by which such processes operate. Rather than treating a person's number of neighbors as the crucial parameter, consider instead the number of distinct social contexts that these neighbors represent as the driving mechanism of social contagion.

The role of neighborhood diversity in contagion processes suggests interesting further directions to pursue, both for mathematical modeling and for potential broader applications. Mathematical models in areas including interacting particle systems [101, 46] and threshold contagion [45, 114] have explored some of the global phenomena that arise from contagion processes in networks for which the behavior at a given node has a nontrivial dependence on the full set of behaviors at neighboring nodes. Neighborhood diversity could be naturally incorporated into such models by basing the underlying contagion probability, for example, on the number of connected components formed by a node's affected neighbors. It then becomes a basic question to understand how the global properties of these processes change when such factors are incorporated.

More broadly, across a range of further domains, these findings suggest an alternate perspective for recruitment to political causes, the promotion of health practices, and marketing; to convince individuals to change their behavior, it may be less important that they receive many endorsements than that they receive the message from multiple directions. In this way, our findings propose a potential revision of core theories for the roles that networks play across social and economic domains.

2.5 Data collection

2.5.1 Recruitment data collection

Here we discuss details of the e-mail recruitment data. All user data were analyzed in an anonymous, aggregated form. The contact neighborhood individuals included in invitation e-mails are limited to nine in number, and so we have restricted our analysis to neighborhoods (inviter plus contact importers) of 10 nodes or less. In cases with more than nine candidate "other people you may know," the invitation tool selects a randomized subset of nine for inclusion in the e-mail.

We conditioned our data collection upon several criteria. First, we considered only first invitations to join the site. Subsequent invitations to an e-mail address are handled differently by the invitation tool, and so we have not included them in our study. Second, we considered only invitations where the inviter invited at most 20 e-mail addresses on the date of the invitation. This conditioning is meant to omit invitation batches where the inviter opted to ?select all? within the contact import tool and focuses our investigation on socially selective invitations.

Invitations were sent during an 11-week period spanning July 12, 2010 to September 26, 2010. An e-mail address was considered to have converted to a registered user account if the address was registered for an account within 14 days of the invitation, counting both individuals who signed up via links provided in the invitation e-mail and users who signed up by visiting the Facebook website directly within 14 days. Only contact import events that occurred before the invitation event are considered. Likewise, only friendship edges that existed before the invitation event are considered to be part of the neighborhood.

Many of the findings we investigate are governed by complex nonlinear effects, which make traditional regression controls generally inadequate. In an attempt to control

for confounding signals in our data, several parallel observation groups were maintained, against which all findings were validated. As a means of capturing potential artifacts from duplicitous private/business e-mail address use, a first such validation group was constructed by conditioning upon e-mail invitations sent to a small set of common and commonly private e-mail providers: Hotmail, Yahoo!, Gmail, AOL, and Yahoo! France. As a means of observing any differences between already established and growing Facebook markets, two parallel validation groups were constructed to observe established markets (United States) and emerging Facebook markets (Brazil, Germany, Japan, and Russia), classified by the most recently resolved country of login for the inviting Facebook account. Whereas invitation conversion rates were generally higher in emerging markets, none of the conditional datasets were observed to deviate from the complete dataset with regard to internal structural findings.

Highly sparse neighborhoods were a very common occurrence in these data, owing to the fact that the neighborhoods we study here are only partial observations of an individual's actual connection to Facebook. We are able to infer links only to those site users who have used the contact importer tool and maintain active e-mail communication with the e-mail address in question, criteria that induce a sampled subgraph that we then observe. The probability of sampling an edge uniformly at random in any neighborhood with low edge density is therefore quite low, and the probability that all sampled nodes come from the same cluster within a clustered neighborhood is lower still. From the perspective of communication multiplexity [70], we should in fact expect that our randomly induced subgraph sample is biased toward strongly connected ties that tend to communicate on multiple mediums, but this expectation is not at issue with our results. The real matter of the fact is that contact neighborhoods where the induced subgraph consists of a single connected component are likely to come from *very* tightly connected neighborhood graphs.

Although the contact importer tool and invitation tool are prominently featured as part of the new user experience on Facebook, they are also heavily used by experienced users of the site: The median site age of an inviter in our dataset was 262 days. Although e-mail invitations constitute only a small portion of Facebook's growth, they provide a valuable window into the otherwise invisible growth process of the Facebook product.

For the analysis of photo co-tags, only co-tags since January 1, 2010 were considered.

2.5.2 Engagement data collection

We consider users *engaged* at a given time point if they have interacted with the application during at least 6 of the last 7 days. As with any measure of user behavior, this metric is a heuristic merely meant to approximate a broader notion of involvement on the site. Highly engaged users who do not access the Internet on weekends will never qualify as "six-plus engaged," whereas users who simply log in on a daily basis to check their messages will qualify. Our analysis is restricted to the population level, so such confounders are not a problem.

Due to the technical nature of how engagement data are stored at Facebook, it is impractical to retrieve six-plus engagement measures for dates exactly 3 months after registration. As an appropriate surrogate, we consider the six-plus engagement of users on the first day of their third calendar month as users.

2.6 The k-brace

In this section we present formal results regarding the notion of a k-brace defined in this work. Recall from the main text that the k-brace is constructed by repeatedly deleting edges of embeddedness less than k until there are no such edges remaining, followed by

a single pass in which all isolated nodes are deleted. The first thing we prove is that this procedure leads to a well-defined outcome. Indeed, some iterative update procedures of this general flavor can potentially produce different end results depending on the order in which the updates are performed; what we wish to show is that the final subgraph produced by the k-brace procedure does not in fact depend on the order in which the edge deletions are performed. To do this, we provide a succinct graph-theoretic characterization of this final subgraph, and then show that all ways of scheduling the edge deletions lead to this subgraph. Finally, we give an efficient algorithm, adapted from the work of Cohen [40], for computing the k-brace.

To characterize the end result of the edge deletion process, we begin with the following definition. Given a graph $G = (V, E)$, a subgraph H of G is a pair (W, F), where $W \subseteq V$ and $F \subseteq E$, and each edge in F has both endpoints in W. We now define the following collection of subgraphs $\mathcal{B}_k(G)$: we say that a subgraph H of G belongs to $\mathcal{B}_k(G)$ if (i) each edge of H belongs to at least k distinct triangles in H, and (ii) each node of H has at least one incident edge in H. We observe that $\mathcal{B}_k(G)$ is a non-empty set, since the subgraph consisting of no nodes and no edges satisfies conditions (i) and (ii), and hence belongs to $\mathcal{B}_k(G)$.

To motivate the definition of property $\mathcal{B}_k(G)$, note that the outcome of the procedure defining the k-brace of G produces a subgraph in $\mathcal{B}_k(G)$. We would like to show more, namely that the k-brace is in fact the unique maximal element of $\mathcal{B}_k(G)$ under a certain natural partial order. In particular, for two subgraphs of G, denoted $H_1 = (W_1, F_1)$ and $H_2 = (W_2, F_2)$, we say that $H_1 \preceq H_2$ if $W_1 \subseteq W_2$ and $F_1 \subseteq F_2$. We now claim the following proposition.

Proposition 1. *In the set $\mathcal{B}_k(G)$, partially ordered by $\preceq$, there is a unique maximal element.*

Proof. Let us define the following union operation on subgraphs: if $H_1 =$

$(W_1, F_1), H_2 = (W_2, F_2), \ldots, H_s = (W_s, F_s)$ are subgraphs of G, then we define their union $\cup_{i=1}^{s} H_i$ to be the subgraph $\left(\cup_{i=1}^{s} W_i, \cup_{i=1}^{s} F_i\right)$.

The key fact underlying the proof is that if $H_1 = (W_1, F_1), H_2 = (W_2, F_2), \ldots, H_s = (W_s, F_s)$ are subgraphs in $\mathcal{B}_k(G)$, then $\cup_{i=1}^{s} H_i$ also belongs to $\mathcal{B}_k(G)$. To see why, we simply observe that (i) every edge in $\cup_{i=1}^{s} H_i$ belongs to at least one of the H_i, and hence is part of at least k triangles; and (ii) every node in $\cup_{i=1}^{s} H_i$ belongs to at least one of the H_i, and hence is incident to at least one edge.

Given this, if we enumerate all the subgraphs $H_1, H_2, \ldots, H_t$ in $\mathcal{B}_k(G)$, then their union $\cup_{i=1}^{t} H_i$ is also an element of $\mathcal{B}_k(G)$. It is the unique maximal element of $\mathcal{B}_k(G)$, since for any subgraph H in $\mathcal{B}_k(G)$, the subgraph H is one of the elements in the union $\cup_{i=1}^{t} H_i$, and hence $H \preceq \cup_{i=1}^{t} H_i$. □

Let $\beta_k(G)$ denote the unique maximal element of $\mathcal{B}_k(G)$. We now claim the following.

Proposition 2. *Any execution of the procedure defining the k-brace, regardless of the order of edge deletions, results in the subgraph $\beta_k(G)$.*

Proof. Consider an execution of the edge deletion procedure, removing edges in the order $e_1, e_2, \ldots, e_s$. Let $G_j = (V, E - \{e_1, e_2, \ldots, e_{j-1}\})$ be the subgraph of G after the first $j-1$ edge deletions, at the moment just before e_j was deleted.

We claim that none of the deleted edges $e_1, e_2, \ldots, e_s$ belong to any subgraph in $\mathcal{B}_k(G)$. Indeed, suppose by way of contradiction that this were not the case, and consider the first edge e_j that does belong to a subgraph $H = (W, F)$ in $\mathcal{B}_k(G)$. In $H = (W, F)$, the edge e_j belongs to a set of k distinct triangles; let $T \subseteq F$ be the set of $2k$ edges other than e_j that constitute these triangles. None of the edges $e_1, e_2, \ldots, e_{j-1}$ can belong to T, since by assumption e_j is the first edge in the sequence of deletions to belong to any subgraph in $\mathcal{B}_k(G)$. But this means that all the edges of T were still present in the underlying graph G_j at the moment that e_j was considered for deletion, and since e_j

therefore belonged to at least k distinct triangles in G_j, it should not have been deleted — a contradiction.

Similarly, we claim that none of the isolated nodes deleted at the end of the procedure belong to any subgraph in $\mathcal{B}_k(G)$. Again, suppose by way of contradiction that one of the deleted nodes v belonged to a subgraph $H = (W, F)$ in $\mathcal{B}_k(G)$. In H, node v is incident to some edge e. But e was not present when v was deleted, and hence e itself must have been deleted earlier in the procedure; hence, $e \in \{e_1, e_2, \ldots, e_s\}$. But we have just shown that none of the edges in $\{e_1, e_2, \ldots, e_s\}$ belong to any subgraph in $\mathcal{B}_k(G)$, whereas e belongs to H, a contradiction.

Finally, consider any execution of the edge deletion procedure, and let H^* denote the subgraph that results from it. H^* belongs to $\mathcal{B}_k(G)$, since at the termination of the procedure all edges in H^* have embeddedness at least k and there are no isolated nodes, and hence by Proposition 1, $H^* \preceq \beta_k(G)$. On the other hand, we have just established that any node or edge that belongs to any subgraph in $\mathcal{B}_k(G)$ also belongs to H^*, and hence $\beta_k(G) \preceq H^*$. It follows that $H^* = \beta_k(G)$, as desired. □

Finally, we describe the following efficient implementation of the edge deletion procedure for computing the k-brace, adapted from Cohen [40].

Algorithm 1 (Extracting the k-brace)**.** *Given a graph G and a parameter k, use a queue q to efficiently traverse the graph and iteratively remove all edges with embeddedness $< k$.*

```
for e ∈ G.edges() do
   Em(e) ← size(G.neighbors(e[0]) ∩ G.neighbors(e[1]));
   if Em(e) < k then
      q.append(e);
      G.removeEdge(e);
   end if
end for
while size(q) != 0 do
   e ← q.pop();
```

```
    I ← G.neighbors(e[0]) ∩ G.neighbors(e[1])
    for v ∈ I do
      e′ ← (e[0],v);
      Em(e′) ← Em(e′) - 1;
      if Em(e′) < k then
        q.append(e′);
        G.removeEdge(e′);
      end if
      e″ ← (e[1],v);
      Em(e″) ← Em(e″) - 1;
      if Em(e″) < k then
        q.append(e″);
        G.removeEdge(e″);
      end if
    end for
  end while
  for v in G.nodes() do
    if degree(v) == 0 then
      G.removeNode(v);
    end if
  end for
```

For a graph with n nodes and m edges, a straight-forward analysis shows that the runtime for this algorithm is at most $O(\sum_{v \in V} \text{degree}^2(v)) = O(m^2)$, which is rather expensive, but fortunately our focus on neighborhood graphs implies that all the graphs we consider are very modest in size.

As mentioned in the text, the k-brace is always a subgraph of the $(k+1)$-core. Since finding the $(k+1)$-core takes merely $O(n+m)$, in practice it is more efficient to first compute the $(k+1)$-core of a graph G, and then find the k-brace of the $(k+1)$-core rather than the full graph G; the analogue of this optimization is also present in Cohen's work [40].

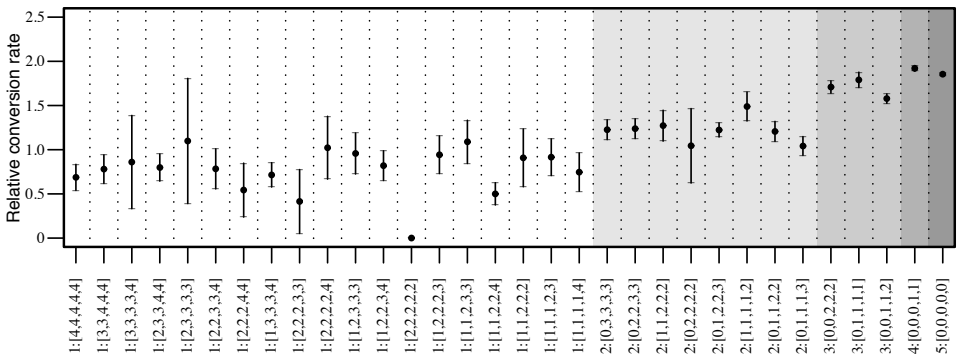

Figure 2.6: Invitation conversion rates of size five neighborhoods, grouped by their number of connected components and their degree distribution, which equates to 31 equivalence classes (for graphs of size four or smaller, degree distribution is a unique determinant of isomorphism, but this is not true for graphs of size five or larger). For example, the label "1:[4,4,4,4,4]" indicates the clique, 1 component where all the nodes have degree 4. The label "2:[1,1,2,2,2]" indicates a graph of two components (a triangle and a pair), while the label "1:[1,1,2,2,2]" indicates the one component line graph. The five-cycle topology "1:[2,2,2,2,2]" was exceedingly rare, and no conversions for this topology were observed. The conversion scale is the same as for the recruitment figures in the main text. Error bars are 95% confidence intervals.

2.7 Additional analyses of recruitment

2.7.1 Five node neighborhood topologies

As an extension of Figure 2.1b-d, we present the recruitment rates for invitation neighborhoods consisting of five nodes, see Figure 2.6. We note that when studying neighborhoods with more than five nodes, the recorded data is spread thinly across an overwhelming number of possible graph topologies, and considering every topology is no longer possible.

2.7.2 Structural vs. demographic diversity

As a potential confounder for our findings, we consider the fact that neighborhoods with many components are comparatively likely to also exhibit increased demographic diversity, which may figure into conversion in a manner outside our structural analysis. To

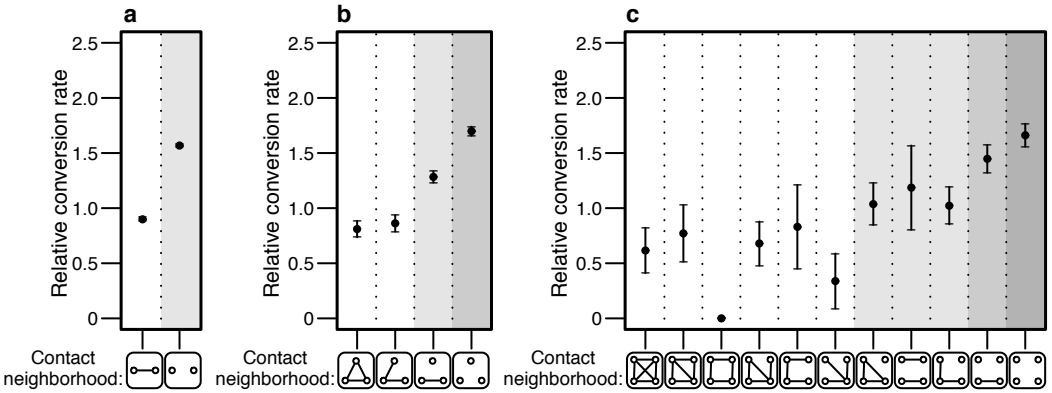

Figure 2.7: Recruitment conversion for demographically homogeneous neighborhoods, as a function of (a) 2-node, (b) 3-node, and (c) 4-node contact neighborhood graphs. The conversion scale is the same as for the recruitment figures in the main text. Error bars represent 95% confidence intervals.

control for this, to the extent that it is possible, we condition our data on neighborhoods that are demographically homogenous with respect to self-reported gender, geography, and age, meaning that all the site users within the neighborhood are of the same gender, from the same country, and all contained within a 5-year range of age. We note that for neighborhoods larger than two in size, this homogeneity requirement entails an aggressive restriction on the amount of admissible data, to the point that for neighborhoods composed of a 4-node cycle, we observe no converted registrations. We find that the significance of our structural measure of neighborhood diversity persists in this demographically controlled dataset; see Figure 2.7.

2.7.3 Embeddedness and weak ties

Here we study the role of Granovetter's structural measure of weak and strong ties, termed *embeddedness*. For an individual $i \in V$ on a social graph $G = (V, E)$, let their neighborhood graph N_i be the subgraph of G induced by their neighboring nodes $V_i = \{j \in V : e_{ij} \in E\}$. *Weak ties*, in a structural sense, are ties with low embeddedness in the social graph, where the *embeddedness* of edge e_{ij} is the number of common

neighbors of the two node endpoints, $Em(e_{ij}) = |N_i \cap N_j|$. As an equivalent definition, the embeddedness of edge e_{ij} is also equal to the degree of node v_j within the neighborhood of node v_i, $Em(e_{ij}) = \deg_{N_i}(j)$. Through this, we observe that the *embeddedness distribution* of a neighborhood, $Em(N_i) = \{Em(e_{ij}) : j \in N_i\}$, is the same as the degree distribution of the neighborhood.

Granovetter's work on 'the strength of weak ties' found that unembedded edges — those with embeddedness zero, termed *local bridges* — play an important role in the spread of awareness for new opportunities, specifically in the labor market [63]. Applying this principle of information novelty to our recruitment domain suggests that invitations arriving along edges with low embeddedness may be more likely to result in successful recruitment. As a consequence, if i is a node who accepts an invitation, one might expect that at least some neighbors j of i will tend to be connected via edges e_{ij} of low embeddedness. In other words, the embeddedness distribution $Em(N_i)$ will have small values: viewed as a multiset, it will contain small numbers as elements.

This leads to a potential confounding effect in our analysis of connected components, in the following way. As noted above, the embeddedness distribution of the neighborhood N_i is the same as its degree distribution; hence, small values in this distribution are consistent with a sparse structure for N_i, and hence with the potential for N_i to have many components. What if the relationship between the number of components and the probability of recruitment is in fact a consequence of the relationship between small numbers in the embeddedness distribution and the probability of recruitment?

Fortunately, we can separate these effects quite cleanly, as follows. There exist pairs of graphs on five and six nodes with precisely the same degree distribution, but with different numbers of connected components (Figure 2.8a). If we look for invitees i whose contact neighborhoods come from these pairs, we will have neighborhoods whose degree distributions — and hence whose embeddedness distributions — are identical,

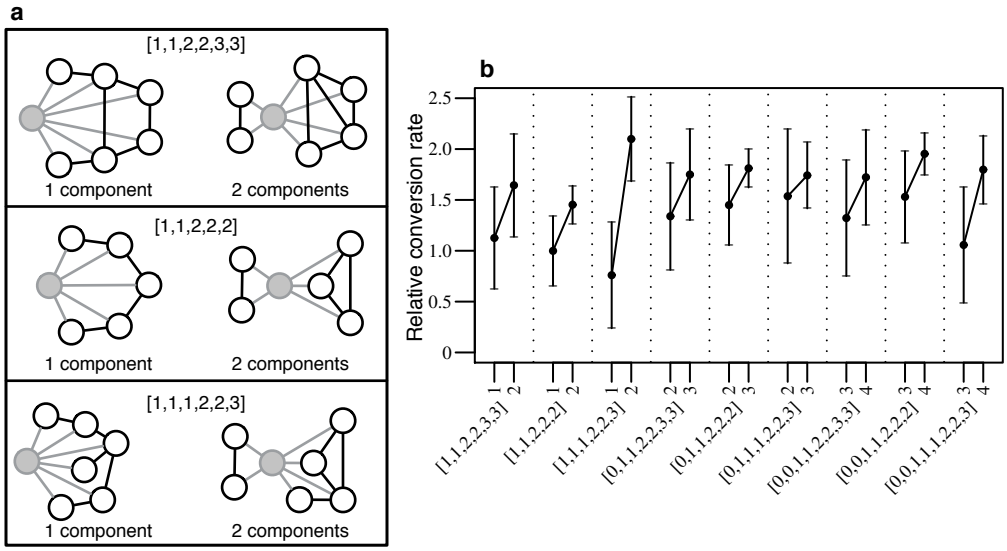

Figure 2.8: Recruitment conversion rates for the nine most frequent pairs of graphs with matched embeddedness distributions. (a) Illustrations of the three most frequent neighborhood graphs pairs with identical embeddedness distributions but differing component counts. The invited node is shown in gray, and the embeddedness distribution refers to the embeddedness of the gray edges. The other six frequent graph topology pairs all contain one of these pairs, up to the addition of a node singleton. (b) The importance of diversity when controlling for embeddedness, examining the nine most common neighborhood graph pairs with identical embeddedness distributions but differing component counts. Each data point is labelled by its degree distribution and its connected component count, as in Figure 2.6. The conversion scale is the same as for the recruitment figures in the main text. Error bars are 95% confidence intervals.

but which have different numbers of connected components. Any argument based on embeddedness values has no way to distinguish among these pairs of graphs, and hence would necessarily predict equivalent rates of recruitment.

Analyzing recruitment rates on precisely these 5-node and 6-node topologies pushes the resolution limits of what is possible even with huge amounts of data, but even so we see that for every such pair of graphs in which the embeddedness distributions are identical but the compoment counts differ, the neighborhood with more components has a higher rate of recruitment (Figure 2.8b). Thus diversity, as measured by component count, appears to play an important role in recruitment conversion in a manner decidedly outside traditional theories of information diffusion.

2.8 Additional analyses of engagement

2.8.1 Predicted engagement for other neighborhood sizes

Here we present results that extend Figure 2.4d-f from the main text, see Figure 2.9. Similar to our comments in the main text, notice that when comparing neighborhoods of different sizes, we can see that having a 30-node neighborhood with two components in the 1-brace predicts as much engagement as having a 50-node neighborhood with only one components in the 1-brace.

2.8.2 Controlling for k-brace size

Figure 2.10 presents a control of the potential confounding factor that k-braces with multiple components are may have a tendency to be larger. Here we control for the size of the 1-brace to show that predicted engagement is still an increasing function of component count when controlling for size.

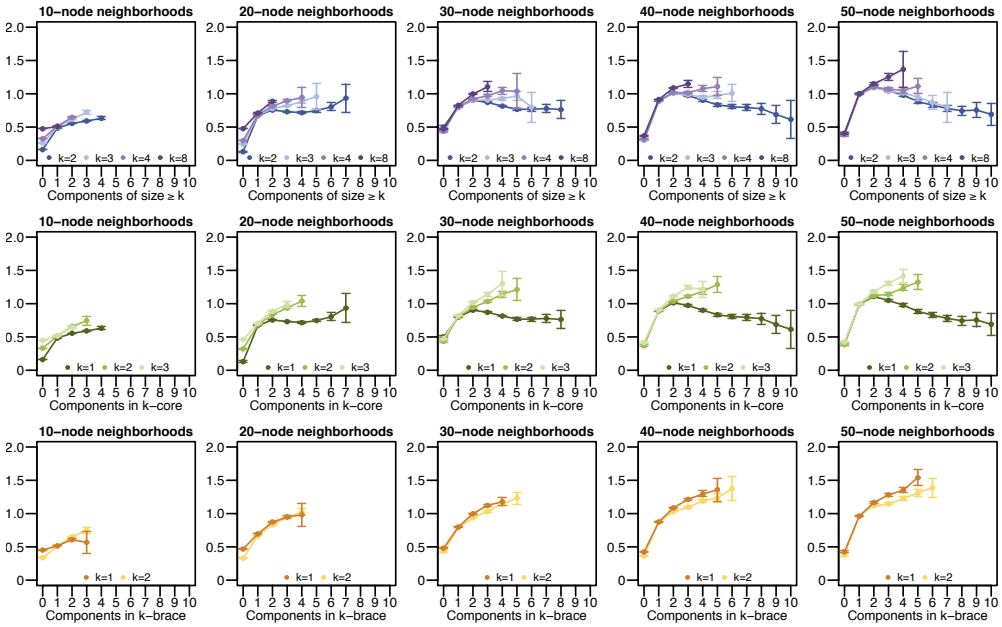

Figure 2.9: Engagement as a function of diversity in a neighborhood, conditioned on size. For each size $n = 10, 20, 30, 40, 50$, plots are shown that correspond to Figure 2.4(d-f), showing the relative engagement rate as a function of component counts. The right three plots correspond exactly to the plots in Figure 2.4(d-f). All engagement rates are reported on a single relative scale, where 1.0 signifies the average conversion rate across all 50-node neighborhoods. Error bars are 95% confidence intervals.

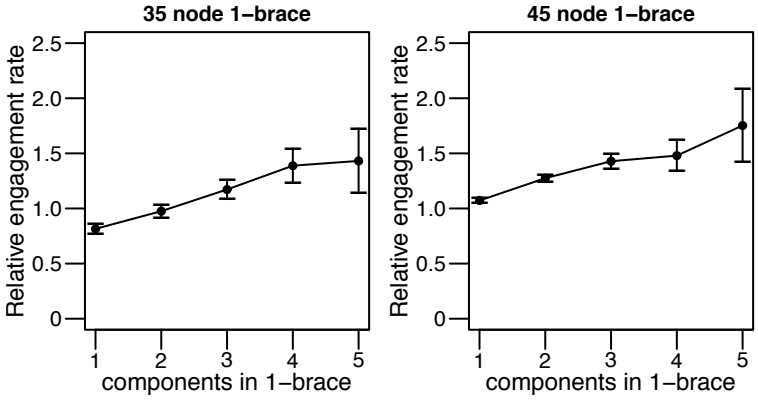

Figure 2.10: Controlling for the size of the k-brace. We focus on neighborhoods of size 50 with exactly 35 and 45 nodes in their 1-brace, and again see that engagement is an increasing function of 1-brace component count. All engagement rates are reported on a single relative scale, where 1.0 signifies the average conversion rate across all 50-node neighborhoods. Error bars are 95% confidence intervals.

CHAPTER 3

SUBGRAPH FREQUENCIES: MAPPING THE EMPIRICAL AND EXTREMAL GEOGRAPHY OF LARGE GRAPH COLLECTIONS

A growing set of on-line applications are generating data that can be viewed as very large collections of small, dense social graphs — these range from sets of social groups, events, or collaboration projects to the vast collection of graph neighborhoods in large social networks. A natural question is how to usefully define a domain-independent 'coordinate system' for such a collection of graphs, so that the set of possible structures can be compactly represented and understood within a common space. In this work, we draw on the theory of graph homomorphisms to formulate and analyze such a representation, based on computing the frequencies of small induced subgraphs within each graph. We find that the space of subgraph frequencies is governed both by its combinatorial properties — based on extremal results that constrain all graphs — as well as by its empirical properties — manifested in the way that real social graphs appear to lie near a simple one-dimensional curve through this space.

We develop flexible frameworks for studying each of these aspects. For capturing empirical properties, we characterize a simple stochastic generative model, a single-parameter extension of Erdős-Rényi random graphs, whose stationary distribution over subgraphs closely tracks the one-dimensional concentration of the real social graph families. For the extremal properties, we develop a tractable linear program for bounding the feasible space of subgraph frequencies by harnessing a toolkit of known extremal graph theory. Together, these two complementary frameworks shed light on a fundamental question pertaining to social graphs: what properties of social graphs are 'social' properties and what properties are 'graph' properties?

We conclude with a brief demonstration of how the coordinate system we examine can also be used to perform classification tasks, distinguishing between structures

arising from different types of social graphs.

3.1 Introduction

The standard approach to modeling a large on-line social network is to treat it as a single graph with an enormous number of nodes and a sparse pattern of connections. Increasingly, however, many of the key problems encountered in managing an on-line social network involve working with large collections of small, dense graphs contained within the network.

On Facebook, for example, the set of people belonging to a group or attending an event determines such a graph, and considering the set of all groups or all events leads to a very large number of such graphs. On any social network, the network neighborhood of each individual — consisting of his or her friends and the links among them — is also generally a small dense graph with a rich structure, on a few hundred nodes or fewer [151]. If we consider the neighborhood of each user as defining a distinct graph, we again obtain an enormous collection of graphs. Indeed, this view of a large underlying social network in terms of its overlapping node neighborhoods suggests a potentially valuable perspective on the analysis of the network: rather than thinking of Facebook, for example, as a single billion-node network, with a global structure that quickly becomes incomprehensible, we argue that it can be useful to think of it as the superposition of a billion small dense graphs — the network neighborhoods, one centered at each user, and each accessible to a closer and more tractable investigation.

Nor is this view limited to a site such as Facebook; one can find collections of small dense graphs in the interactions within a set of discussion forums [54], within a set of collaborative on-line projects [157], and in a range of other settings.

Our focus in the present work is on a fundamental global question about these types of graph collections: given a large set of small dense graphs, can we study this set by

defining a meaningful 'coordinate system' on it, so that the graphs it contains can be represented and understood within a common space? With such a coordinate system providing a general-purpose framework for analysis, additional questions become possible. For example, when considering collections of a billion or more social graphs, it may seem as though almost any graph is possible; is that the case, or are there underlying properties guiding the observed structures? And how do these properties relate to more fundamental combinatorial constraints deriving from the extremal limits that govern all graphs? As a further example, we can ask how different graph collections compare to one another; do network neighborhoods differ in some systematic way, for instance, from social graphs induced by other contexts, such as the graphs implicit in social groups, organized events, or other arrangements?

The Present Work In this chapter we develop and analyze such a representation, drawing on the theory of *graph homomorphisms*. Roughly speaking, the coordinate system we examine begins by describing a graph by the frequencies with which all possible small subgraphs occur within it. More precisely, we choose a small number k (e.g. $k = 3$ or 4); then, for each graph G in a collection, we create a vector with a coordinate for each distinct k-node subgraph H, specifying the fraction of k-tuples of nodes in G that induce a copy of H (in other words, the frequency of H as an induced subgraph of G). For $k = 3$, this description corresponds to what is sometimes referred to as the *triad census* [44, 50, 51, 158]. The literature on *frequent subgraph mining* [74, 91, 161], and *motif counting* [109] is also is closely related, but focuses on connected subgraphs.

With each graph in the collection mapped to such a vector, we can ask how the full collection of graphs fills out this space of subgraph frequencies. This turns out to be a subtle issue, because the arrangement of the graphs in this space is governed by two distinct sets of effects: extremal combinatorial constraints showing that certain combinations of subgraph frequencies are genuinely impossible; and empirical properties,

which reveal that the bulk of the graphs tend to lie close to a simple one-dimensional curve through the space. We formulate results on both these types of properties, in the former case building on an expanding body of combinatorial theory [25, 103] for bounding the frequencies at which different types of subgraphs can occur in a larger ambient graph.

The fact that the space of subgraph frequencies is constrained in these multiple ways also allows us to concretely address the following type of question: When we see that human social networks do not exhibit a certain type of structure, is that because such a structure is mathematically impossible, or simply because human beings do not create it when they form social connections? In other words, what is a property of graphs and what is a property of people? Although this question is implicit in many studies of social networks, it is hard to separate the two effects without a formal framework such as we have here.

Indeed, our framework offers a direct contribution to one of the most well-known observations about social graphs: the tendency of social relationships to close triangles, and the relative infrequency of what is sometimes called the 'forbidden triad': three people with two social relationships between them, but one absent relationship [128]. There are many sociological theories for why one would expect this subgraph to be underrepresented in empirical social networks [63]. Our framework shows that the frequency of this 'forbidden triad' has a non-trivial upper bound in not just social graphs, but in all graphs. Harnessing our framework more generally, we are in fact able to show that *any* k node subgraph that is not a complete or empty subgraph has a frequency that is bounded away from one. Thus, there is an extent to which almost all subgraphs are mathematically 'forbidden' from occurring beyond a certain frequency.

We aim to separate these mathematical limits of graphs from the complementary empirical properties of real social graphs. The fact that real graph collections have a

roughly one-dimensional structure in our coordinate system leads directly to our first main question: is it possible to succinctly characterize the underlying backbone for this one-dimensional structure, and can we use such a characterization to usefully describe graphs within our coordinate system in terms of their deviation from this backbone?

The subgraph frequencies of the standard Erdős-Rényi random graph [22] $G_{n,p}$ produce a one-dimensional curve (parametrized by p) that weakly approximates the layout of the real graphs in the space, but the curve arising from this random graph model systematically deviates from the real graphs in that the random graph contains fewer triangles and more triangle-free subgraphs. This observation is consistent with the sociological principle of *triadic closure* — that triangles tend to form in social networks. As a means of closing this deviation from $G_{n,p}$, we develop a tractable stochastic model of graph generation with a single additional parameter, determining the relative rates of arbitrary edge formation and triangle-closing edge formation. The model exhibits rich behaviors, and for appropriately chosen settings of its single parameter, it produces remarkably close agreement with the subgraph frequencies observed in real data for the suite of all possible 3-node and 4-node subgraphs.

Finally, we use this representation to study how different collections of graphs may differ from one another. This arises as a question of basic interest in the analysis of large social media platforms, where users continuously manage multiple audiences [13] — ranging from their set of friends, to the members of a groups they've joined, to the attendees of events and beyond. Do these audiences differ from each other at a structural level, and if so what are the distinguishing characteristics? Using Facebook data, we identify structural differences between the graphs induced on network neighborhoods, groups, and events. The underlying basis for these differences suggests corresponding distinctions in each user's reaction to these different audiences with whom they interact.

3.2 Data description

Throughout our presentation, we analyze several collections of graphs collected from Facebook's social network. The collections we study are all induced graphs from the Facebook friendship graph, which records friendship connections as undirected edges between users, and thus all our induced graphs are also undirected. The framework we characterize in this work would naturally extend to provide insights about directed graphs, an extension we do not discuss. We do not include edges formed by Facebook 'subscriptions' in our study, nor do we include Facebook 'pages' or connections from users to such pages. All Facebook social graph data was analyzed in an anonymous, aggregated form.

For this work, we extracted three different collections of graphs, around which we organize our discussion:

- **Neighborhoods**: Graphs induced by the friends of a single Facebook user *ego* and the friendship connections among these individuals (excluding the ego).
- **Groups**: Graphs induced by the members of a 'Facebook group', a Facebook feature for organizing focused conversations between a small or moderate-sized set of users.
- **Events**: Graphs induced by the confirmed attendees of 'Facebook events', a Facebook feature for coordinating invitations to calendar events. Users can response 'Yes', 'No', and 'Maybe' to such invitations, and we consider only users who respond 'Yes'.

The neighborhood and groups collections were assembled in October 2012 based on monthly active user egos and current groups, while the events data was collected from all events during 2010 and 2011. For event graphs, only friendship edges formed prior to the date of the event were used. Subgraph frequencies for four-node subgraphs were computed by sampling 11,000 induced subgraphs uniformly with replacement, providing sufficiently precise frequencies without enumeration. The graph collections

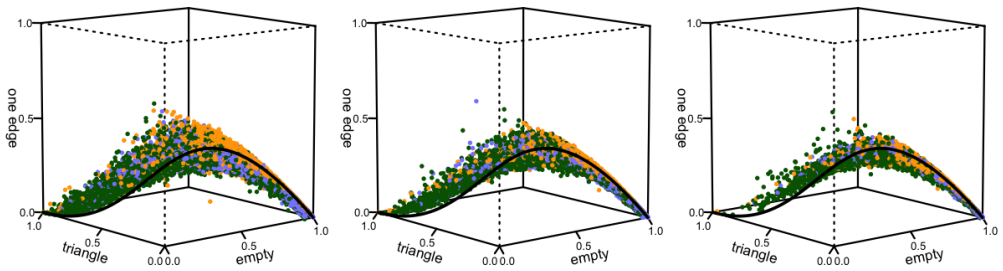

Figure 3.1: Subgraph frequencies for three node subgraphs for graphs of size 50, 100, and 200 (left to right). The neighborhoods are orange, groups are green, and events are lavender. The black curves illustrate $G_{n,p}$ as a function of p.

were targeted at a variety of different graph sizes, as will be discussed in the text.

3.3 Subgraph space

In this section, we study the space of subgraph frequencies that form the basis of our coordinate system, and the one-dimensional concentration of empirical graphs within this coordinate system. We derive a model capable of accurately identifying the backbone of this empirical concentration using only the basic principle of *triadic closure*, showing how the subgraph frequencies of empirical social graphs are seemingly restricted to the vicinity of a simple one-dimensional structure.

Formally, the subgraph frequency of a k-node graph F in an n-node graph G (where $k \leq n$) is the probability that a random k-node subset of G induces a copy of F. It is clear that for any integer k, the subgraph frequencies of all the k-node graphs sum to one, constraining the vector of frequencies to an appropriately dimensioned simplex. In the case of $k = 3$, this vector is simply the relative frequency of induced three-node subgraphs restricted to the 4-simplex; there are just four such subgraphs, with zero, one, two, and three edges respectively. When considering the frequency of larger subgraphs, the dimension of the simplex grows very quickly, and already for $k = 4$, the space of

four-node subgraph frequencies lives in an 11-simplex.

Empirical distribution In Figure 3.1, the three-node subgraph frequencies of 50-node, 100-node, and 200-node graph collections are shown, with each subplot showing a balanced mixture of 17,000 neighborhood, group and event graphs – the three collections discussed in Section 3.2, totaling 51,000 graphs at each size. Because these frequency vectors are constrained to the 4-simplex, their distribution can be visualized in $\mathbb{R}^3$ with three of the frequencies as axes.

Notice that these graph collections, induced from disparate contexts, all occupy a sharply concentrated subregion of the unit simplex. The points in the space have been represented simply as an unordered scatterplot, and two striking phenomena already stand out: first, the particular concentrated structure within the simplex that the points follow; and second, the fact that we can already discern a non-uniform distribution of the three contexts (neighborhoods, groups and events) within the space — that is, the different contexts can already be seen to have different structural loci. Notice also that as the sizes of the graphs increases – from 50 to 100 to 200 – the distribution appears to sharpen around the one-dimensional backbone. The vast number of graphs that we are able to consider by studying Facebook data is here illuminating a structure that is simply not discernible in previous examinations of subgraph frequencies [51], since no analysis has previously considered a collection near this scale.

The imagery of Figure 3.1 directly motivates our work, by visually framing the essence of our investigation: what facets of this curious structure derive from our graphs being social graphs, and what facets are simply universal properties of all graphs? We will find, in particular, that parts of the space of subgraph frequencies are in fact inaccessible to graphs for purely combinatorial reasons — it is mathematically impossible for one of the points in the scatterplot to occupy these parts of the space. But there are other parts of the space that are mathematically possible; it is simply that no real social

graphs appear to be located within them. Intuitively, then, we are looking at a population density within an ambient space (the Facebook graphs within the space of subgraph frequencies), and we would like to understand both the geography of the inhabited terrain (what are the properties of the areas where the population has in fact settled?) and also the properties of the boundaries of the space as a whole (where, in principle, would it be possible for the population to settle?).

Also in Figure 3.1, we plot the curve for the frequencies for 3 node subgraphs in $G_{n,p}$ as a function of p. The curves are given simply by the probability of obtaining the desired number of edges in a three node graph, $((1-p)^3, 3p(1-p)^2, 3p^2(1-p), p^3)$. This curve closely tracks the empirical density through the space, with a single notable discrepancy: the real world graphs systemically contain more triangles when compared to $G_{n,p}$ at the same edge density. We emphasize that it is not a priori clear why $G_{n,p}$ would at all be a good model of subgraph frequencies in modestly-sized dense social graphs such as the neighborhoods, groups, and events that we have here; we believe the fact that it tracks the data with any fidelity at all is an interesting issue for future work. Beyond $G_{n,p}$, in the following subsection, we present a stochastic model of edge formation and deletion on graphs specifically designed to close the remaining discrepancy. As such, our model provides a means of accurately characterizing the backbone of subgraph frequencies for social graphs.

Stochastic model of edge formation The classic Erdős-Rényi model of random graphs, $G_{n,p}$, produces a distribution over n-node undirected graphs defined by a simple parameter p, the probability of each edge independently appearing in the graph. We now introduce and analyze a related random graph model, the *Edge Formation Random Walk*, defined as a random walk over the space of all unlabeled n-node graphs. In its simplest form, this model is closely related to $G_{n,p}$, and will we show via detailed balance that the distribution defined by $G_{n,p}$ on n-node graphs is precisely the stationary

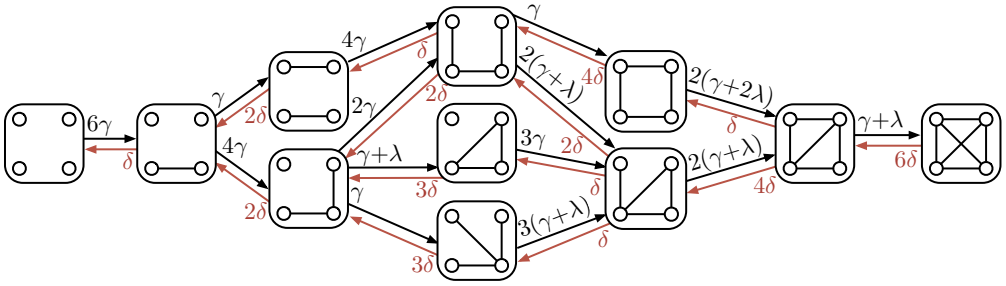

Figure 3.2: The state transitions diagram for our stochastic graph model with $k = 4$, where γ is the arbitrary edge formation rate, λ is the triadic closure formation rate, and δ is the edge elimination rate.

distribution of this simplest version of the random walk on the space of n-node graphs. We first describe this basic version of the model; we then add a component to the model that captures a triadic closure process, which produces a close fit to the properties we observe in real graphs.

Let $\mathcal{G}_n$ be the space of all unlabeled n-node graphs, and let $X(t)$ be the following continuous time Markov chain on the state space $\mathcal{G}_n$. The transition rates between the graphs in $\mathcal{G}_n$ are defined by random additions and deletions of edges, with all edges having a uniform formation rate $\gamma > 0$ and a uniform deletion rate $\delta > 0$. Thus the single parameter $\nu = \gamma/\delta$, the effective formation rate of edges, completely characterizes the process. Notice that this process is clearly irreducible, since it is possible to transition between any two graphs via edge additions and deletions.

Since $X(t)$ is irreducible, it possesses a unique stationary distribution. The stationary distribution of an irreducible continuous time Markov chain can be found as the unique stable fixed point of the linear dynamical system $X'(t) = Q_n(\nu)X(t)$ that describes the diffusion of probability mass during a random walk on n-node graphs, where $Q_n(\nu)$ is the generator matrix with transition rates q_{ij} and $q_{ii} = -\sum_{j \neq i} q_{ji}$, all depending only on ν. The stationary distribution π_n then satisfies $Q_n(\nu)\pi_n = 0$.

The following proposition shows the clear relationship between the stationary dis-

tribution of this simplest random walk and the frequencies of $G_{n,p}$.

Proposition 3.3.1. *The probabilities assigned to (unlabeled) graphs by $G_{n,p}$ satisfy the detailed balance condition for the Edge Formation Random Walk with edge formation rate $\nu = p/1-p$, and thus characterizes the stationary distribution.*

Proof. We first describe an equivalent Markov chain based on labeled graphs: there is a state for each labeled n-node graph; the transition rate q_{ij} from a labelled graph G_i to a labelled graph G_j is $q_{ij} = \gamma$ if G_j can be obtained from G_i by adding an edge; and $q_{ij} = \delta$ if G_j can be obtained from G_i by removing an edge. All other transition rates are zero. We call this new chain the *labeled chain*, and the original chain the *unlabeled chain.*

Now, suppose there is a transition from unlabeled graph H_a to unlabeled graph H_b in the unlabeled chain, with transition probability $k\gamma$. This means that there are k ways to add an edge to a labeled copy of H_a to produce a graph isomorphic to H_b. Now, let G_i be any graph in the labeled chain that is isomorphic to H_a. In the labeled chain, there are k transitions out of G_i leading to a graph isomorphic to H_b, and each of these has probability γ. Thus, with probability $k\gamma$, a transition out of G_i leads to a graph isomorphic to H_b. A strictly analogous argument can be made for edge deletions, rather than edge additions.

This argument shows that the following describes a Markov chain equivalent to the original unlabeled chain: we draw a sequence of labeled graphs from the labeled chain, and we output the isomorphism classes of these labeled graphs. Hence, to compute the stationary distribution of the original unlabeled chain, which is what we seek, we can compute the stationary distribution of the labeled chain and then sum stationary probabilities in the labeled chain over the isomorphism classes of labeled graphs.

It thus suffices to verify the detailed balance condition for the distribution on the labeled chain that assigns probability $p^{|E(G_i)|}(1-p)^{\binom{n}{2}-|E(G_i)|}$ to each labeled graph G_i.

Since every transition of the labeled walk occurs between two labeled graphs G_i and G_j, with $|E(G_i)| = |E(G_j)| + 1$, the only non-trivial detailed balance equations are of the form:

$$\begin{aligned} q_{ij}\text{Pr}[X(t) = G_i] &= q_{ji}\text{Pr}[X(t) = G_j] \\ \text{Pr}[X(t) = G_i] &= \nu\text{Pr}[X(t) = G_j] \\ \text{Pr}[X(t) = G_i] &= \frac{p}{1-p}\text{Pr}[X(t) = G_j]. \end{aligned}$$

Since the probability assigned to the labeled graph G_i by $G_{n,p}$ is simply $p^{|E(G_i)|}(1 - p)^{\binom{n}{2}-|E(G_i)|}$, detailed balance is clearly satisfied. □

Incorporating triadic closure The above modeling framework provides a simple analog of $G_{n,p}$ that notably exposes itself to subtle adjustments. By simply adjusting the transition rates between select graphs, this framework makes it possible to model random graphs where certain types of edge formations or deletions have irregular probabilities of occurring, simply via small perturbations away from the classic $G_{n,p}$ model. Using this principle, we now characterize a random graph model that differs from $G_{n,p}$ by a single parameter, λ, the rate at which 3-node paths in the graph tend to form triangles. We call this model the *Edge Formation Random Walk with Triadic Closure*.

Again let $\mathcal{G}_n$ be the space of all unlabeled n-node graphs, and let $Y(t)$ be a continuous time Markov chain on the state space $\mathcal{G}_n$. As with the ordinary Edge Formation Random Walk, let edges have a uniform formation rate $\gamma > 0$ and a uniform deletion rate $\delta > 0$, but now also add a triadic closure formation rate $\lambda \geq 0$ for every 3-node path that a transition would close. The process is still clearly irreducible, and the stationary distribution obeys the stationary conditions $Q_n(\nu, \lambda)\pi_n = 0$, where the generator matrix Q_n now also depends on λ. We can express the stationary distribution directly in the parameters as $\pi_n(\nu, \lambda) = \{\pi : Q_n(\nu, \lambda)\pi = 0\}$. For $\lambda = 0$ the model reduces to the ordinary Edge Formation Random Walk.

The state transitions of this random graph model are easy to construct for $n = 3$ and $n = 4$, and transitions for the case of $n = 4$ are shown in Figure 3.2. Proposition 3.3.1 above tells us that for $\lambda = 0$, the stationary distribution of a random walk on this state space is given by the graph frequencies of $G_{n,p}$. As we increase λ away from zero, we should therefore expect to see a stationary distribution that departs from $G_{n,p}$ precisely by observing more graphs with triangles and less graphs with open triangles.

The framework of our Edge Formation Random Walk makes it possible to model triadic closure precisely; in this sense the model forms an interesting contrast with other models of triangle-closing in graphs that are very challenging to analyze (e.g. [36, 76, 94, 126, 144]). We will now show how the addition of this single parameter makes it possible to describe the subgraph frequencies of empirical social graphs with remarkable accuracy.

Fitting subgraph frequencies The stationary distribution of an Edge Formation Random Walk model describes the frequency of different graphs, while the coordinate system we are developing focuses on the frequency of k-node subgraphs within n-node graphs. For $G_{n,p}$ these two questions are in fact the same, since the distribution of random induced k-node subgraphs of $G_{n,p}$ is simply $G_{k,p}$. When we introduce $\lambda > 0$, however, our model departs from this symmetry, and the stationary probabilities in a random walk on k node graphs is no longer precisely the frequencies of induced k-node subgraphs in a single n-node graph.

But if we view this as a model for the frequency of small graphs as objects in themselves, rather than as subgraphs of a larger ambient graph, the model provides a highly tractable parameterization that we can use to approximate the structure of subgraph frequencies observed in our families of larger graphs. In doing so, we aim to fit $\pi_k(\nu(p, \lambda), \lambda)$ as a function of p, where $\nu(p, \lambda)$ is the rate parameter ν that produces edge density p for the specific value of λ. For $\lambda = 0$ this relationship is simply $\nu = p/(1-p)$,

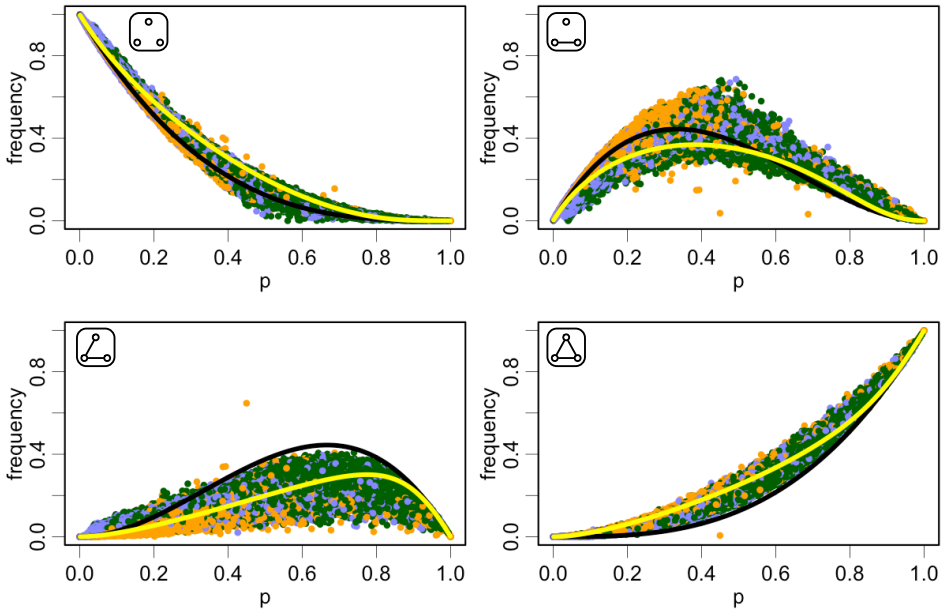

Figure 3.3: Subgraph frequencies for 3-node subgraphs in 50-node graphs, shown as a function of p. The black curves illustrate $G_{n,p}$, while the yellow curves illustrate the fit model.

but for $\lambda > 0$ the relation is not so tidy, and in practice it is easier to fit ν numerically rather than evaluate the expression.

When considering a collection of graph frequencies we can fit λ by minimizing residuals with respect to the model. Given a collection of N graphs, let $y_k^1, \ldots, y_k^N$ be the vectors of k-node subgraph frequencies for each graph and $p^1, \ldots, p^N$ be the edge densities. We can then fit λ as:

$$\lambda_k^{opt} = \arg\min_{\lambda} \sum_{i=1}^{N} ||\pi_k(\nu(p^i, \lambda), \lambda) - y_k^i||_2.$$

In Figure 3.3 we plot the three-node subgraph frequencies as a function of edge density p, for a collection of 300,000 50-node subgraphs, again a balanced mixture of neighborhoods, groups, and events. In this figure we also plot (in yellow) the curve resulting from fitting our random walk model with triadic closure, $\pi_k(\nu(p, \lambda_k^{opt}), \lambda_k^{opt})$, which is thus parameterized as a function of edge density p. For this mixture of collections and $k = 3$, the optimal fit is $\lambda_3^{opt} = 1.61$. Notice how the yellow line deviates

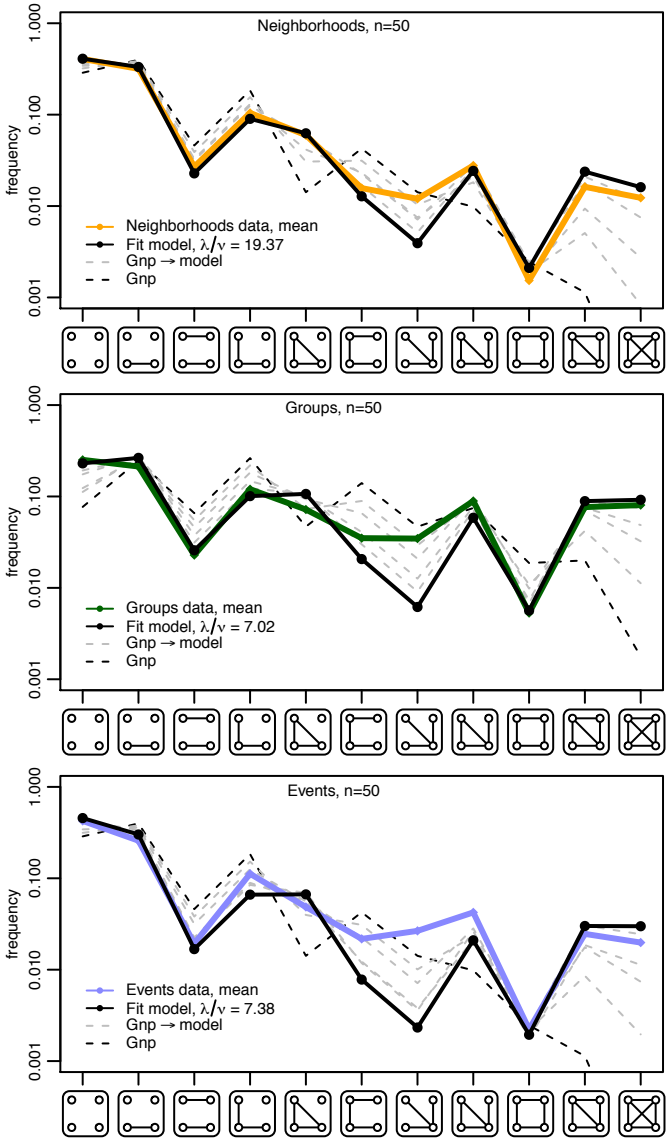

Figure 3.4: The four-node subgraph frequencies for the means of the 50-node graph collections in Figure 3.3, and the subgraph frequency of the model, fitting the triadic closure rate λ to the mean vectors. As λ increases from $\lambda = 0$ to $\lambda = \lambda_{opt}$, we see how this single additional parameter provides a striking fit.

from the black $G_{n,p}$ curve to better represent the backbone of natural graph frequencies. From the figure it is clear that almost all graphs have more triangles than a sample from $G_{n,p}$ of corresponding edge density. When describing extremal bounds in Section 3.4, we will discuss how $G_{n,p}$ is in fact by no means the extremal lower bound.

As suggested by Figure 3.2, examining the subgraph frequencies for four-node subgraphs is fully tractable. In Figure 3.4, we fit λ to the mean subgraph frequencies of our three different collections of graphs separately. Note that the mean of the subgraph frequencies over a set of graphs is not necessarily itself a subgraph frequency corresponding to a graph, but we fit these mean 11-vectors as a demonstration of the model's ability to fit an 'average' graph. The subgraph frequency of $G_{n,p}$ at the edge density corresponding to the data is shown as a black dashed line in each plot — with poor agreement — and gray dashed lines illustrate an incremental transition in λ, starting from zero (when it corresponds to $G_{n,p}$) and ending at λ^{opt}.

The striking agreement between the fit model and the mean of each collection is achieved at the corresponding edge density by fitting only λ. For neighborhood graphs, this agreement deviates measurably on only a single subgraph frequency, the four-node star. The y-axis is plotted on a logarithmic scale, which makes it rather remarkable how precisely the model describes the scarcity of the four-node cycle. The scarcity of squares is observed in email neighborhoods on Facebook in Chapter 2, and our model provides the first intuitive explanation of this scarcity.

The model's ability to characterize the backbone of the empirical graph frequencies suggests that the subgraph frequencies of individual graphs can be usefully studied as deviations from this backbone. In fact, we can interpret the fitting procedure for λ as a variance minimization procedure. Recall that the mean of a set of points in $\mathbb{R}^n$ is the point that minimizes the sum of squared residuals. In this way, the procedure is in fact fitting the 'mean curve' of the model distribution to the empirical subgraph frequencies.

Finally, our model can be used to provide a measure of the triadic closure strength differentially between graph collections, investigating the difference in λ^{opt} for the subgraph frequencies of different graph collections. In Figure 3.4, the three different graph types resulted in notably different ratios of λ/ν — the ratio of the triadic closure for-

mation rate to the basic process rate — with a significantly higher value for this ratio in neighborhoods. We can interpret this as saying that open triads in neighborhoods are more prone to triadic closure than open triads in groups or events.

3.4 Extremal bounds

As discussed at the beginning of the previous section, we face two problems in analyzing the subgraph frequencies of real graphs: to characterize the distribution of values we observe in practice, and to understand the combinatorial structure of the overall space in which these empirical subgraph frequencies lie. Having developed stochastic models to address the former question, we now consider the latter question.

Specifically, in this section we characterize extremal bounds on the set of possible subgraph frequencies. Using machinery from the theory of graph homomorphisms, we identify fundamental bounds on the space of subgraph frequencies that are not properties of social graphs, but rather, are universal properties of all graphs. By identifying these bounds, we make apparent large tracts of the feasible region that are theoretically inhabitable but not populated by any of the empirical social graphs we examine.

We first review a body of techniques based in extremal graph theory and the theory of graph homomorphisms [103]. We use these techniques to formulate a set of inequalities on subgraph frequencies; these inequalities are all linear for a fixed edge density, an observation that allows us to cleanly construct a linear program to maximize and minimize each subgraph frequency within the combined constraints. In this manner, we show how it is possible to map outer bounds on the geography of all these structural constraints. We conclude by offering two basic propositions that transcend all edge densities, thus identifying fundamental limits on subgraph frequencies of all sizes.

3.4.1 Background on subgraph frequency and homomorphism density

In this subsection, we review some background arising from the theory of graph homomorphisms. We will use this homomorphism machinery to develop inequalities governing subgraph frequencies. These inequalities allow us to describe the outlines of the space underlying Figure 3.1(a) — the first step in understanding which aspects of the distribution of subgraph frequencies in the simplex are the result of empirical properties of human social networks, and which are the consequences of purely combinatorial constraints.

Linear constraints on subgraph frequency Let $s(F, G)$ denote the subgraph frequency of F in G, as defined in the last section: the probability that a random $|V(F)|$-node subset of G induces a copy of F. Note that since $s(F, G)$ is a probability over outcomes, it is subject to the law of total probability. The law of total probability for subgraph frequencies takes the following form.

Proposition 3.4.1. *For any graph F and any integer $\ell \geq k$, where $|V(F)| = k$, the subgraph density of F in G, $s(F, G)$ satisfies the equality*

$$s(F, G) = \sum_{\{H:|V(H)|=\ell\}} s(F, H)s(H, G).$$

Proof. Let H' be a random ℓ-vertex induced subgraph of G. Now, the set of outcomes $\mathcal{H} = \{H : |V(H)| = \ell\}$ form a partition of the sample space, each with probability $s(H, G)$. Furthermore, conditional upon an ℓ-vertex induced subgraph being isomorphic to H, $s(F, H)$ is the probability that a random k-vertex induced subgraph of H is isomorphic to F. □

This proposition characterizes an important property of subgraph frequencies: the vector of subgraph frequencies on k nodes exists in a linear subspace of the vector of

subgraph frequencies on $\ell > k$ nodes. Furthermore, this means that any constraint on the frequency of a subgraph F will also constrain the frequency of any subgraph H for which $s(F, H) > 0$ or $s(H, F) > 0$.

Graph homomorphisms A number of fundamental inequalities on the occurrence of subgraphs are most naturally formulated in terms of *graph homomorphisms*, a notion that is connected to but distinct from the notion of induced subgraphs. In order to describe this machinery, we first review some basic definitions [25]. if F and G are labelled graphs, a map $f : V(F) \to V(G)$ is a *homomorphism* if each edge (v, w) of F maps to an edge $(f(v), f(w))$ of G. We now write $t(F, G)$ for the probability that a random map from $V(F)$ into $V(G)$ is a homomorphism, and we refer to $t(F, G)$ as a *homomorphism density* of F and G.

There are three key differences between the homomorphism density $t(F, G)$ and the subgraph frequency $s(F, G)$ defined earlier in this section. First, $t(F, G)$ is based on mappings of F into G that can be many-to-one — multiple nodes of F can map to the same node of G — while $s(F, G)$ is based on one-to-one mappings. Second, $t(F, G)$ is based on mappings of F into G that must map edges to edges, but impose no condition on pairs of nodes in F that do not form edges: in other words, a homomorphism is allowed to map a pair of unlinked nodes in F to an edge of G. This is not the case for $s(F, G)$, which is based on maps that require non-edges of F to be mapped to non-edges of G. Third, $t(F, G)$ is a frequency among mappings from labeled graphs F to labelled graphs G, while $s(F, G)$ is a frequency among mappings from unlabeled F to unlabeled G.

From these three differences, it is not difficult to write down a basic relationship governing the functions s and t [25]. To do this, it is useful to define the intermediate notion $t_{\text{inj}}(F, G)$, which is the probability that a random *one-to-one* map from $V(F)$ to $V(G)$ is a homomorphism. Since only an $O(1/V(G))$ fraction of all maps from $V(F)$

to $V(G)$ are not one-to-one, we have

$$t(F, G) = t_{\text{inj}}(F, G) + O(1/|V(G)|). \tag{3.1}$$

Next, by definition, a one-to-one map f of F into G is a homomorphism if and only if the image $f(F)$, when viewed as an induced subgraph of G, contains all of F's edges and possibly others. Correcting also for the conversion from labelled to unlabeled graphs, we have

$$t_{\text{inj}}(F, G) = \sum_{F' : F \subseteq F'} \frac{\text{ext}(F, F') \cdot \text{aut}(F')}{k!} \cdot s(F', G), \tag{3.2}$$

where $\text{aut}(F')$ is the number of automorphisms of F' and $\text{ext}(F, F')$ is the number of ways that a labelled graph F can be extended (by adding edges) to form a labelled graph H isomorphic to F'.

Homomorphism inequalities There are a number of non-trivial results bounding the graph homomorphism density, which we now review. By translating these to the language of subgraph frequencies, we can begin to develop bounds on the simplexes in Figure 3.1.

For complete graphs, the Kruskal-Katona Theorem produces upper bounds on homomorphism density in terms of the edge density while the Moon-Moser Theorem provides lower bounds, also in terms of the edge density.

Proposition 3.4.2 (Kruskal-Katona [103])**.** *For a complete graph K_r on r nodes and graph G with edge density $t(K_2, G)$,*

$$t(K_r, G) \leq t(K_2, G)^{r/2}.$$

Proposition 3.4.3 (Moon-Moser [112, 130])**.** *For a complete graph K_r on r nodes and graph G with edge density $t(K_2, G) \in [(k-2)/(k-1), 1]$,*

$$t(K_r, G) \geq \prod_{i=1}^{r-1} (1 - i(1 - t(K_2, G))).$$

The Moon-Moser bound is well known to not be sharp, and Razborov has recently given an impressive sharp lower bound for the homomorphism density of the triangle K_3 [130] using sophisticated machinery [129]. We limit our discussion to the simpler Moon-Moser lower bound which takes the form of a concise polynomial and provides bounds for arbitrary r, not just the triangle ($r = 3$).

Finally, we employ a powerful inequality that is known to lower bound the homomorphism density of any graph F that is either a forest, an even cycle, or a complete bipartite graph. Stated as such, it is the solved special cases of the open *Sidorenko Conjecture*, which posits that the result could be extended to all bipartite graphs F. We will use the following proposition in particular when F is a tree, and will refer to this part of the result as the *Sidorenko tree bound*.

Proposition 3.4.4 (Sidorenko [103, 137])**.** *For a graph F that is a forest, even cycle, or complete bipartite graph, with edge set E(F), and G with edge density $t(K_2, G)$,*

$$t(F, G) \geq t(K_2, G)^{|E(F)|}.$$

Using Equations (3.1) and (3.2), we can translate statements about homomorphisms into asymptotic statements about the combined frequency of particular sets of subgraphs. We can also translate statements about frequencies of subgraphs to frequencies of their complements using the following basic fact.

Lemma 3.4.5. *If for graphs $F_1, \ldots F_\ell$, coefficients $\alpha_i \in \mathbb{R}$, and a function f,*

$$\alpha_1 s(F_1, G) + \ldots + \alpha_\ell s(F_\ell, G) \geq f(s(K_2, G)),\ \forall G,$$

then

$$\alpha_1 s(\overline{F}_1, G) + \ldots + \alpha_\ell s(\overline{F}_\ell, G) \geq f(1 - s(K_2, G)),\ \forall G.$$

Proof. Note that $s(F, G) = s(\overline{F}, \overline{G})$. Thus if

$$\alpha_1 s(\overline{F}_1, \overline{G}) + \ldots + \alpha_\ell s(\overline{F}_\ell, \overline{G}) \geq f(s(\overline{K}_2, \overline{G})),\ \forall G,$$

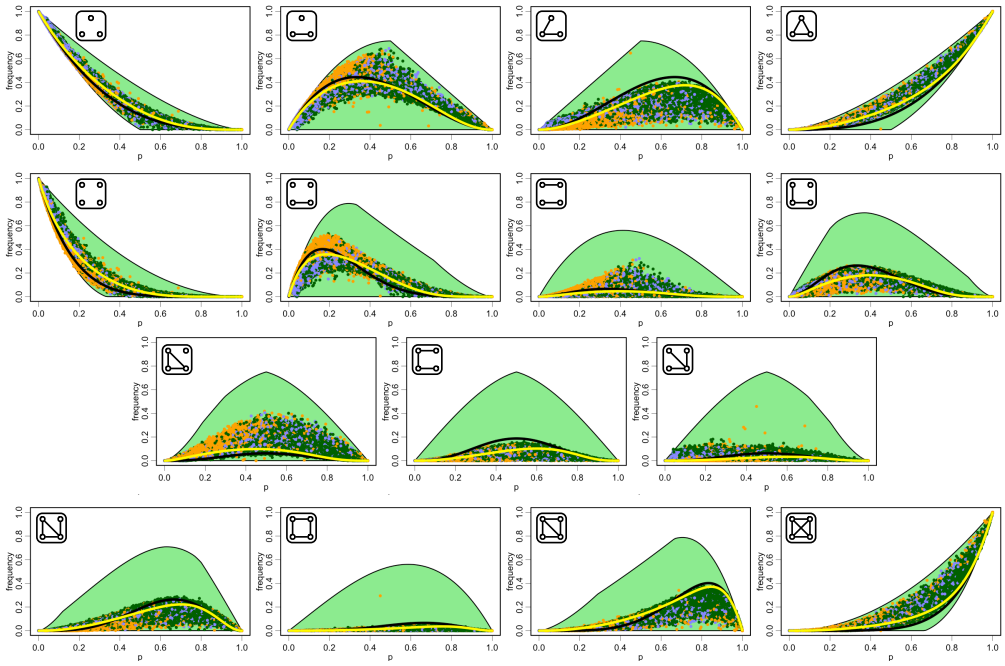

Figure 3.5: Subgraph frequencies for 3-node and 4-node subgraphs as function of edge density p. The light green regions denote the asymptotically feasible region found via the linear program. The empirical frequencies are as in Figure 3.3. The black curves illustrate $G_{n,p}$, while the yellow curves illustrate the fit triadic closure model.

then

$$\alpha_1 s(\overline{F}_1, G) + \ldots + \alpha_\ell s(\overline{F}_\ell, G) \geq f(s(\overline{K}_2, G)),\ \forall G,$$

where $s(\overline{K}_2, G) = 1 - s(K_2, G)$. □

3.4.2 An LP for subgraph frequency bounds

In the previous section, we reviewed linear constraints between the frequencies of subgraphs of different sizes, and upper and lower bounds on graph homomorphism densities with applications to subgraph frequencies. We will now use these constraints to assemble a linear program capable to mapping out bounds on the extremal geography of the subgraph space we are considering. To do this, we will maximize and minimize the frequency of each individual subgraph frequency, subject to the constraints we have just

catalogued.

We will focus our analysis on the cases $k = 3$, the triad frequencies, and $k = 4$, the quad frequencies. Let x_1, x_2, x_3, x_4 denote the subgraph frequencies $s(\cdot, G)$ of the four possible 3-vertex undirected graphs, ordered by increasing edge count.

Program 3.4.6. *The frequency x_i of a 3-node subgraph in any graph G with edge density p is bounded asymptotically (in $|V(G)|$) by $\max / \min x_i$ subject to $x_i \geq 0, \forall i$ and:*

$$x_1 + x_2 + x_3 + x_4 = 1, \qquad \frac{1}{3}x_2 + \frac{2}{3}x_3 + x_4 = p, \tag{3.3}$$

$$x_4 \leq p^{3/2}, \qquad x_1 \leq (1-p)^{3/2}, \tag{3.4}$$

$$x_4 \geq p(2p-1) \qquad p \geq 1/2, \tag{3.5}$$

$$x_1 \geq (1-p)(1-2p) \qquad p \leq 1/2, \tag{3.6}$$

$$(1/3)x_3 + x_4 \geq p^2, \qquad x_1 + (1/3)x_2 \geq (1-p)^2. \tag{3.7}$$

Here the equalities in (3.3) derive from the linear constraints, the constraints in (3.4) derive from Kruskal-Katona, the constraints (3.5-3.6) derive from Moon-Moser, and the constraints in (3.7) derive from the Sidorenko tree bound. More generally, we obtain the following general linear program that can be used to find nontrivial bounds for any subgraph frequency:

Program 3.4.7. *The frequency f_F of a k-node subgraph F in any graph G with edge density p is bounded asymptotically (in $|V(G)|$) by $\max / \min f_F$, subject to $Af_F = b(p), Cf_F \leq d(p)$, appropriately assembled.*

From Program 1 given above it is possible to derive a simple upper bound on the frequency of the 3-node-path (sometimes described in the social networks literature as the "forbidden triad", as mentioned earlier).

Proposition 3.4.8. *The subgraph frequency of the 3-node-path F obeys $s(F, G) \leq 3/4 + o(1), \forall G$.*

Proof. Let x_1, x_2, x_3, x_4 again denote the subgraph frequencies $s(\cdot, G)$ of the four possible 3-vertex undirected graphs, ordered by increasing edge count, where x_3 is the frequency of the 3-node-path. By the linear constraints,

$$(1/3)x_2 + (2/3)x_3 + x_4 = p,$$

while by Moon-Moser, $x_4 + O(1/|V(G)|) \geq p(2p-1)$. Combining these two constraints we have:

$$x_3 \leq 3p(1-p) + o(1).$$

The polynomial in p is maximized at $p = 1/2$, giving an upper bound of $3/4 + o(1)$. □

This bound on the "forbidden triad" is immediately apparent from Figure 3.5 as well, which shows the bounds constructed via linear programs for all 3-node and 4-node subgraph frequencies. In fact, the subgraph frequency of the 'forbidden" 3-node-path in the balanced complete bipartite graph $K_{n/2,n/2}$, which has edge density $p = 1/2$, is exactly $s(F, G) = 3/4$, demonstrating that this bound is asymptotically tight. (In fact, we can perform a more careful analysis showing that it is exactly tight for even n.)

Figure 3.5 illustrates these bounds for $k = 3$ and $k = 4$. Notice that our empirical distributions of subgraph frequencies fall well within these bounds, leaving large tracts of the bounded area uninhabited by any observed dense social graph. While the bounds do not fully characterize the feasible region of subgraph frequencies, the fact that the bound is asymptotically tight at $p = 1/2$ for the complete bipartite graph $K_{n/2,n/2}$ is important — practically no empirical social graphs come close to the boundary, despite this evidence that it is feasibly approachable. We emphasize that an exact characterization of the feasible space would necessitate machinery at least as sophisticated as that used by Razborov.

In the next subsection we develop two more general observations about the subgraph frequencies of arbitrary graphs, the latter of which illustrates that, with the exception of

clique subgraphs and empty subgraphs, it is always possible to be free from a subgraph. This shows that the lower regions of the non-clique non-empty frequency bounds in Figure 3.5 are always inhabitable, despite the fact that social graphs do not empirically populate these regions.

3.4.3 Bounding frequencies of arbitrary subgraphs

The upper bound for the frequency of the 3-node-path given in Proposition 3.4.8 amounted to simply combining appropriate upper bounds for different regions of possible edge densities p. In this section, we provide two general bounds pertaining to the subgraph frequency of an arbitrary subgraph F. First, we show that any subgraph that is not a clique and is not empty must have a subgraph density bounded strictly away from one. Second, we show that for every subgraph F that is not a clique and not empty, it is always possible to construct a family of graphs with any specified asymptotic edge density p that contains no induced copies of F.

With regard to Figures 3.5, the first of the results in this subsection uses the Sidorenko tree bound to show that in fact no subgraph other than the clique or the empty graph, not even for large values of k, has a feasible region that can reach a frequency of $1 - o(1)$. The second statement demonstrates that it is always possible to be free of any subgraph that is not a clique or an empty graph, even if this does not occur in the real social graphs we observe.

Proposition 3.4.9. *For every k, there exist constants ε and n_0 such that the following holds. If F is a k-node subgraph that is not a clique and not empty, and G is any graph on $n \geq n_0$ nodes, then $s(F, G) < 1 - \varepsilon$.*

Proof. Let S_k denote the k-node star — in other words the tree consisting of a single node linked to $k - 1$ leaves. By Equation (3.1), if G has n nodes, then $t_{\text{inj}}(S_k, G) \geq$

$t(S_k, G) - c/n$ for an absolute constant c. We now state our condition on ε and n_0 in the statement of the proposition: we choose ε small enough and n_0 large enough so that

$$\frac{(1-\varepsilon)^k}{2\binom{k}{2}^{k-1}} > \max\left(\varepsilon, \frac{c}{n}\right). \tag{3.8}$$

For a k-node graph F, let $\mathcal{P}(F)$ denote the property that for all graphs G on at least n_0 nodes, we have $s(F, G) < 1 - \varepsilon$. Our goal is to show that $\mathcal{P}(F)$ holds for all k-node F that are neither the clique nor the empty graph. We observe that since $s(F, G) = s(\overline{F}, \overline{G})$, the property $\mathcal{P}(F)$ holds if and only if $\mathcal{P}(\overline{F})$ holds.

The basic idea of the proof is to consider any k-node graph F that is neither complete nor empty, and to argue that the star S_k lacks a one-to-one homomorphism into at least one of F or $\overline{F}$ — suppose it is F. The Sidorenko tree bound says that S_k must have a non-trivial number of one-to-one homomorphisms into G; but the images of these homomorphisms must be places where F is not found as an induced subgraph, and this puts an upper bound on the frequency of F.

We now describe this argument in more detail; we start by considering any specific k-node graph F that is neither a clique nor an empty graph. We first claim that there cannot be a one-to-one homomorphism from S_k into both of F and $\overline{F}$. For if there is a one-to-one homomorphism from S_k into F, then F must contain a node of degree $k-1$; this node would then be isolated in $\overline{F}$, and hence there would be no one-to-one homomorphism from S_k into $\overline{F}$. Now, since it is enough to prove that just one of $\mathcal{P}(F)$ or $\mathcal{P}(\overline{F})$ holds, we choose one of F or $\overline{F}$ for which there is no one-to-one homomorphism from S_k. Renaming if necessary, let us assume it is F.

Suppose by way of contradiction that $s(F, G) \geq 1 - \varepsilon$. Let q denote the edge density of F — that is, $q = |E(F)|/\binom{k}{2}$. The edge density p of G can be written, using Proposition 3.4.1, as

$$\begin{aligned} p = s(K_2, G) &= \sum_{\{H : |V(H)| = k\}} s(K_2, H) s(H, G) \\ &\geq s(K_2, F) s(F, G) \geq q(1-\varepsilon). \end{aligned}$$

By a k-*set* of G, we mean a set of k nodes in G. We color the k-sets of G according to the following rule. Let U be a k-set of G: we color U *blue* if $G[U]$ is isomorphic to F, and we color U *red* if there is a one-to-one homomorphism from S_k to $G[U]$. We leave the k-set uncolored if it is neither blue nor red under these rules. We observe that no k-set U can be colored both blue and red, for if it is blue, then $G[U]$ is isomorphic to F, and hence there is no one-to-one homomorphism from S_k into $G[U]$. Also, note that $s(F, G) \geq 1 - \varepsilon$ is equivalent to saying that at least a $(1 - \varepsilon)$ fraction of all k-sets are blue.

Finally, what fraction of k-sets are red? By the Sidorenko tree bound, we have

$$t(S_k, G) \geq p^{k-1} \geq q^k(1-\varepsilon)^k \geq \frac{(1-\varepsilon)^k}{\binom{k}{2}^{k-1}},$$

where the last inequality follows from the fact that F is not the empty graph, and hence $q \geq 1/\binom{k}{2}$. Since $t_{\text{inj}}(S_k, G) \geq t(S_k, G) - c/n$, our condition on n from (3.8) implies that

$$t_{\text{inj}}(S_k, G) \geq \frac{(1-\varepsilon)^k}{2\binom{k}{2}^{k-1}} > \varepsilon.$$

Now, let $\text{inj}(S_k, G)$ denote the number of one-to-one homomorphisms of S_k into G; by definition,

$$t_{\text{inj}}(S_k, G) = \frac{\text{inj}(S_k, G)}{n(n-1)\cdots(n-k+1)} = \frac{\text{inj}(S_k, G)}{k!\binom{n}{k}},$$

and hence

$$\text{inj}(S_k, G) = k!\binom{n}{k} t_{\text{inj}}(S_k, G) > \varepsilon k!\binom{n}{k}.$$

Now, at most $k!$ different one-to-one homomorphisms can map S_k to the same k-set of G, and hence more than $\varepsilon\binom{n}{k}$ many k-sets of G are red. It follows that the fraction of k-sets that are red is $> \varepsilon$; but this contradicts our assumption that at least a $(1 - \varepsilon)$ fraction of k-sets are blue, since no k-set can be both blue and red. □

Proposition 3.4.10. *Assume F is not a clique and not empty. Then for each edge density p there exists a sequence $G_1^p, G_2^p, \ldots$ of asymptotic edge density p for which F does not*

appear as an induced subgraph in any G_i^p. Equivalently, $s(F, G_i^p) = 0, \forall i$.

Proof. We call H a *near-clique* if it has at most one connected component of size greater than one, and this component is a clique. For any $p \in [0, 1]$, it is possible to construct an infinite sequence $H_1^p, H_2^p, \ldots$ of near-cliques with asymptotic density p, by simply taking the non-trivial component of each H_i^p to be a clique of the appropriate size.

Now, fix any $p \in [0, 1]$, and let F be any graph that is neither a clique nor an empty graph. If F is not a near-clique, then the required sequence $G_1^p, G_2^p, \ldots$ is the sequence of near-cliques $H_1^p, H_2^p, \ldots$, since all the induced subgraphs of a near-clique are themselves near-cliques.

On the other hand, if F is a near-clique, then since F is neither a clique nor an empty graph, the complement of F is not a near-clique. It follows that the required sequence $G_1^p, G_2^p, \ldots$ is the sequence of complements of the near-cliques $H_1^{1-p}, H_2^{1-p}, \ldots$. □

Note that it is possible to take an F-free graph with asymptotic density p and append nodes with local edge density p and random (Erdős-Rényi) connections to obtain a graph with any intermediate subgraph frequency between zero and that of $G_{n,p}$. The same blending arguement can be applied to any graph with a subgraph frequency above $G_{n,p}$ to again find graphs with intermediate subgraph frequencies. In this way we see that large tracts of the subgraph frequency simplex are fully feasible for arbitrary graphs, yet by Figure 3.5 are clearly not inhabited by any real world social graph.

3.5 Classification of audiences

The previous two sections characterize empirical and extremal properties of the space of subgraph frequencies, providing two complementary frameworks for understanding the structure of social graphs. In this section, we conclude our work with a demonstration of how subgraph frequencies can also provide a useful tool for distinguishing between

different categories of graphs. The Edge Formation Random Walk model introduced in Section 3.3 figures notably, providing a meaningful baseline for constructing classification features, contributing to the best overall classification accuracy we are able to produce.

Thus, concretely our classification task is to take a social graph and determine whether it is a node neighborhood, the set of people in a group, or the set of people at an event. This is a specific version of a broader characterization problem that arises generally in social media — namely how social audiences differ in terms of social graph structure [3]. Each of the three graph types we discuss — neighborhoods, groups, and events — define an audience with which a user may choose to converse. The defining feature of such audience decisions has typically been their size — as users choose to share something online, do they want to share it publicly, with their friends, or with a select subgroup of their friends? Products such as Facebook groups exist in part to address this audience problem, enabling the creation of small conversation circles. Our classification task is essentially asking: do audiences differ in meaningful structural ways other than just size?

In Figure 3.1 and subsequently in Figure 3.5, we saw how the three types of graphs that we study — neighborhoods, groups, and events — are noticeably clustered around different structural foci in the space of subgraph frequencies. Figure 3.5 focused on graphs consisting of exactly 50-nodes, where it is visibly apparent that both neighborhoods and events tend to have a lower edge density than groups of that size. Neighborhood edge density — equivalent to the *local clustering coefficient* — is known to generally decrease with graph size [121], but it is not clear that all three of the graph types we consider here should decrease at the same rate.

In Figure 3.6, we see that in fact the three graph types do not decrease uniformly, with the average edge density of neighborhoods decreasing more slowly than groups

or events. Thus, small groups are denser than neighborhoods while large groups are sparser, with the transition occurring at around 400 nodes. Similarly, small event graphs are denser than neighborhoods while large events are much sparser, with the transition occurring already at around 75 nodes.

The two crossing points in Figure 6 suggest a curious challenge: are their structural features of audience graphs that distinguish them from each other even when they exhibit the same edge density? Here we use the language of subgraph frequencies to formulate a classification task for classifying audience graphs based on subgraph frequencies. We compare our classification accuracy to the accuracy achieved when also considering a generous vector of much more sophisticated graph features. We approach this classification task using a simple logistic regression model. While more advanced machine learning models capable of learning richer relationships would likely produce better classification accuracies, our goal here is to establish that this vocabulary of features based on subgraph frequencies can produce non-trivial classification results even in conjunction with simple techniques such as logistic regression. Evaluating our features in other contexts such as graph matching [84, 98, 156], where frequencies of connected subgraphs have been used previously [136], would be interesting future work.

When considering neighborhood graphs, recall that we are not including the ego of the neighborhoods as part of the graph, while for groups and events the administrators as members of their graphs. As such, neighborhoods without their ego deviate systematicallly from analogous audience graphs created as groups or as events. In Figure 3.6 we also show the average edge density of neighborhoods with their ego, adding one node and $n - 1$ edges, noting that the difference is small for larger graphs.

Classification features Subgraph frequencies has been the motivating coordinate system for the present work, and will serve as our main feature set. Employing the Edge Formation Random Walk model from Section 3.3, we additionally describe each graph

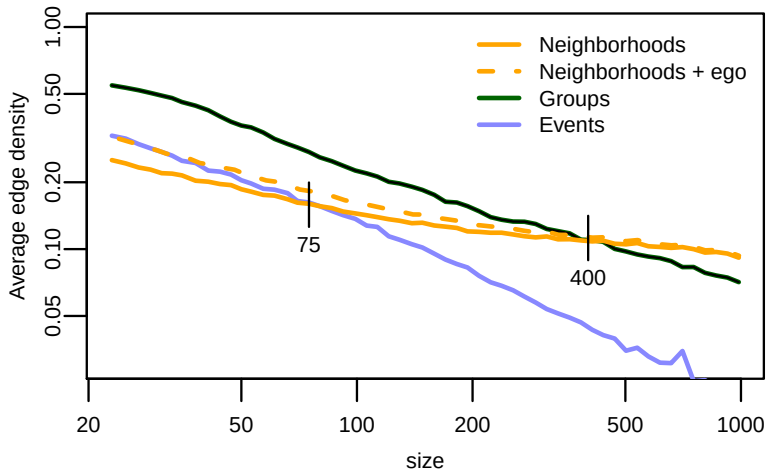

Figure 3.6: Edge densities of neighborhoods, groups, and events as a function of size, n. When $n < 400$, groups are denser then neighborhoods. When $n < 75$, events are denser then neighborhoods.

by its residuals with respect to a backbone — described by the parameter λ — fit to the complete unclassified training set.

Features based on subgraph frequencies are local features, computable by examining only a few local nodes of the graph at a time. Note that the subgraph frequencies of arbitrarily large graphs can be accurately approximated by sampling a small number of induced graphs. Comparatively, it is relevant to ask: can these simple local features do as well as more sophisticated *global* graph features? Perhaps the number of connected components, the size of the largest component, or other global features provide highly informative features for graph classification.

To answer this question, we compare our classification accuracy using subgraph frequencies with the accuracy we are able to achieve using a set of global graph features. We consider:

- Size of the k largest components, for $k = 1, 2$.
- Size of the k-core, for $k = 0, 1, 2, 3$.
- Number of components in the k-core, for $k = 0, 1, 2$.

- Degeneracy, the largest k for which the k-core is non-empty.
- Size of the k-brace (See Chapter 2), for $k = 1, 2, 3$.
- Number of components in the k-brace, for $k = 1, 2, 3$.

These features combine linearly to produce a rich set of graph properties. For example, the number of components in the 1-core minus the number of components in the 0-core yields the number of singletons in the graph.

Classification results The results of the classification model are shown in Table 3.1, reported in terms of classification accuracy — the fraction of correct classifications on the test data – measured using five-fold cross-validation on a balanced set of 10,000 instances. The classification tasks were chosen to be thwart classification based solely on edge density, which indeed performs poorly. Using only 4-node subgraph frequencies and residuals, an accuracy of 77% is achieved in both tasks.

In comparison, classification based on a set of global graph features performed worse, achieving just 69% and 76% accuracy for the two tasks. Meanwhile, combining global and subgraph frequency features performed best of all, with a classification accuracy of $81-82\%$. In each case we also report the accuracy with and without residuals as features. Incorporating residuals with respect to either a $G_{n,p}$ or Edge Formation Random Walk baseline consistently improved classification, and examining residuals with respect to either baseline clearly provides a useful orientation of the subgraph coordinate system for empirical graphs.

3.6 Conclusion

The modern study of social graphs has primarily focused on the examination of the sparse large-scale structure of human relationships. This global perspective has led to

Model Features	N vs. E, $n = 75$	N vs. G, $n = 400$
Edges	0.487	0.482
Triads	0.719	0.647
Triads + R_G	0.737	0.673
Triads + R_λ	0.736	0.668
Quads	0.751	0.755
Quads + R_G	0.765	0.769
Quads + R_λ	0.765	0.769
Global + Edges	0.694	0.763
Global + Triads	0.785	0.766
Global + Triads + R_G	0.784	0.766
Global + Triads + R_λ	0.789	0.767
Global + Quads	0.797	0.812
Global + Quads + R_G	0.807	0.815
Global + Quads + R_λ	0.809	0.820

Table 3.1: Classification accuracy for Neighborhoods (N), Groups (G), and Events (E) on different sets of features. R_G and R_λ denote the residuals with respect to a $G_{n,p}$ and stochastic graph model baseline, as described in the text.

fruitful theoretical frameworks for the study of many networked domains, notably the world wide web, computer networks, and biological ecosystems [121]. However, in this work we argue that the locally dense structure of social graphs admit an additional framework for analyzing the structure of social graphs.

In this work, we examine the structure of social graphs through the coordinate system of subgraph frequencies, developing two complementary frameworks that allow us to identify both 'social' structure and 'graph' structure. The framework developed in Section 3.3 enables us to characterize the apparent social forces guiding graph formation, while the framework developed in Section 3.4 characterizes fundamental limits of all graphs, delivered through combinatorial constraints. Our coordinate system and frameworks are not only useful for developing intuition, but we also demonstrate how they can be used to accurately classify graph types using only these simple descriptions in terms of subgraph frequency.

3.7 Code

Implementations of the Edge Formation Random Walk equilibrium solver and the subgraph frequency bounds optimization program are available from my academic homepage.

CHAPTER 4

BALANCED LABEL PROPAGATION FOR PARTITIONING MASSIVE GRAPHS

Partitioning graphs at scale is a key challenge for any application that involves distributing a graph across disks, machines, or data centers. Graph partitioning is a very well studied problem with a rich literature, but existing algorithms typically can not scale to billions of edges, or can not provide guarantees about partition sizes.

In this work we introduce an efficient algorithm, *balanced label propagation*, for precisely partitioning massive graphs while greedily maximizing edge locality, the number of edges that are assigned to the same shard of a partition. By combining the computational efficiency of label propagation — where nodes are iteratively relabeled to the same 'label' as the plurality of their graph neighbors — with the guarantees of constrained optimization — guiding the propagation by a linear program constraining the partition sizes — our algorithm makes it practically possible to partition graphs with billions of edges.

Our algorithm is motivated by the challenge of performing graph predictions in a distributed system. Because this requires assigning each node in a graph to a physical machine with memory limitations, it is critically necessary to ensure the resulting partition shards do not overload any single machine.

We evaluate our algorithm for its partitioning performance on the Facebook social graph, and also study its performance when partitioning Facebook's 'People You May Know' service (PYMK), the distributed system responsible for the feature extraction and ranking of the friends-of-friends of all active Facebook users. In a live deployment, we observed average query times and average network traffic levels that were 50.5% and 37.1% (respectively) when compared to the previous naive random sharding.

4.1 Introduction

In Plato's *Phaedrus*, Socrates tells us that a key principle of rhetoric is the ability to divide ideas "where the natural joints are, and not trying to break any part, after the manner of a bad carver" [56]. This, too, is the goal of graph partitioning: breaking a graph at its natural joints.

In this work, we present an algorithm for finding 'joints' in graphs of particularly massive proportions, with an emphasis on the Facebook social graph consisting of over 800 million nodes and over 68 billion edges [151]. The algorithm we present uses label propagation to relocate inefficiently assigned nodes while respecting strict shard balancing constraints. We show how this *balanced label propagation algorithm* can be formulated as a convex optimization problem that reduces to a manageable linear programming problem. The algorithm is fundamentally iterative, where each iteration entails solving a linear program.

While we find that our algorithm performs well under random initialization, by initializing the algorithm with a greedy geographic assignment, we find that it is possible to effectively achieve convergence within a single step of the update algorithm, while random initialization requires many iterations to slowly converges to a less performative solution.

Our algorithm has the ability to follow arbitrary partition size specifications, a generalization of the more common goal of symmetric partitioning [78]. This functionality makes it possible to use the greedy efficiency of label propagation when considering partitions that are not symmetric by creation, but might still benefit from label propagation. Specifically, the geographic initialization we consider is not symmetric, yet balanced label propagation can still be used to considerably improve the partitioning.

Label propagation unfortunately offers no formal performance guarantees with respect to the graph partitioning objective, and neither does our modification. It is worth

remarking that the basic problem of constrained graph partitioning, bisecting a graph into two equal parts with as few crossing edges as possible, is the well known NP-hard *minimum bisection problem* [58], with the best known approximation algorithm being a $O(\sqrt{n}\log n)$-factor approximation [52]. Knowing this, our goal has been to develop a highly scalable algorithm that can deliver precise partition size guarantees while performing well at maximizing edge locality in practice.

People You May Know. After presenting our algorithm and analyzing its performance on both the Facebook social graph and a LiveJournal graph, we then present the results of a full deployment of the partitioning scheme for the Facebook friend recommendation system, 'People You May Know' (PYMK). This system computes, for a given user u, and each friend-of-friend (FoF) w of u a feature vector $x_{u,w}$ of graph metrics based on the local structure between u and w. The system then uses machine learning to rank all the suggestions w for u, based on the feature vector $x_{u,w}$, as well as demographic features of u and w.

Due to the size of the graph and high query volume, Facebook has built a customized system for performing this task. Each user u is assigned to a specific machine m_u (this assignment is what we are optimizing). When a FoF query is issued for a user u, it must be sent to machine m_u. Then, for each $v \in N(u)$, a query must be issued to the machine m_v, to retrieve the nodes two hops from u. The results of these queries are then aggregated to compute the features that are input to the machine learning phase, which outputs the final ranked list of FoFs.

Our goal is to perform our graph sharding such that, as often as possible, $m_v = m_u$ for $v \in N(u)$. By doing this well, we can reduce the number of other machines that we need to query, and also reduce the total amount of data that needs to be transferred over the network, increasing overall system throughput and latency. We are constrained by the fact that machines have a limited amount of memory, which puts a hard cap on the

size of each shard.

At the outset, it is important to note that it is not actually clear that a sophisticated sharding will be a performance win in this application. When naively sharding a graph, one typical approach is to assign nodes to machines by simply taking the modulus of the user ID; see Figure 4.1. The advantage of this approach is that the machine location of a node is directly encoded in the user ID. Using a non-trivial sharding requires extensive additional lookups in a *shard map*, as well as an additional network operation at the start of the query where the initial request is forwarded to the appropriate machine hosting the subject of the query. The shard map for all 800 million nodes must also be mirrored in memory across all machines.

As we will see in our ultimate demonstration, the cost of this additional complexity is greatly overshadowed by the improvements in locality that can be had. The novel graph sharding algorithm we introduce, combined with the intelligent initialization based on geographic metadata, is able to produce a sharding across 78 machines where 75.2% of edges are local to individual machines. In the conclusion to this work, we deploy the resulting sharding on Facebook's 'People You May Know' realtime service, and observe dramatic performance gains in a realtime environment.

The main distinction when designing an algorithm applicable at Facebook's scale is that the full graph can not be easily stored in memory. Realistically, this means that the only admissible algorithms are those that examine the graph in streaming iterations. One iteration of our balanced label propagation algorithm takes a single aggregating pass over the edge list, executable in MapReduce, and then solves a linear program with complexity dependent on the number of machines, not the size of the graph. A final assignment step takes a single streaming pass over the nodes, again in MapReduce, to redefine their assignments. Recent work on streaming graph partitioning [142] produced notable performance using a single greedy iteration, but the results were obtained within

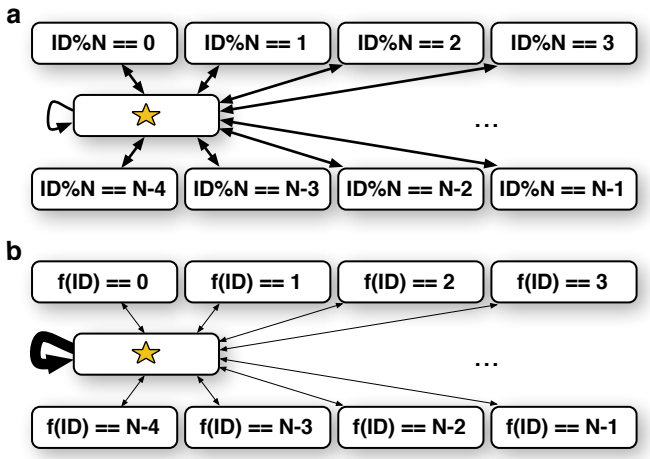

Figure 4.1: Sharding the neighbors of a node across N machines. (a) Aggregating properties of the neighbors when the edge list is sharded according to node ID mod N. (b) Aggregating when the edge list is sharded according to a shardmap f. The goal of an efficient shard map is to greatly increase the likelihood that a node is located on the same shard as its neighbors.

a framework that did not naturally lend itself to additional iteration.

Organization of the chapter. Section 2 presents the details of the balanced label propagation algorithm. Section 3 discusses a geographic initialization that, when possible, greatly improves performance. In Section 4, we evaluate the algorithm and its ability to shard the Facebook social graph and a publicly availably LiveJournal graph dataset. In Section 5, we study a deployment of the sharding to load-balance Facebook's PYMK service. We conclude with Section 6 by discussing future directions for work on this problem.

4.2 Balanced label propagation

Label propagation was first proposed as an efficient method for learning missing labels for graph data in a semi-supervised setting [162]. In such a setting, unlabeled nodes iteratively adopt the label of the plurality of their neighbors until convergence, making

it possible to infer a broad range of traits that are fundamentally assortative along the edges of a graph.

In a context very similar to graph partitioning, label propagation has been found to be a very efficient technique for network community detection [127], the challenge of finding naturally dense network clusters in an unlabeled graph [55]. In this context, each node of a graph is initialized with an individual label, and label propagation is iterated where nodes again update their labels to the plurality of their neighbor's labels.

Network community detection and graph partitioning are very similar challenges, with two key differences. First, community detection algorithms need not and should not require *a priori* specification of the number or size of graph communities to find. Second, community detection algorithms ought to support overlapping communities, while graph partitions seek explicitly disjoint structure. While some partitioning applications may benefit from assigning nodes to multiple partitions, the Facebook social graph we aim to partition has a modest maximum degree of $5,000$, and so we restrict our investigation to creating true partitions.

Label propagation fails to detect overlapping communities [55], though an adaptation does exist [65]. But this failure of the ordinary label propagation algorithm in fact makes it a strong candidate for graph partitioning. The remaining difficulty is therefore that label propagation provides no way of constraining the sizes of any of the resulting community partitions. Our contribution address precisely this difficulty.

A previous attempt to 'constrain' label propagation utilizes an optimization framework with a cost penalty to encourage balanced partitions [18], but this approach does not offer constraints in a formal sense. The constraint-based algorithm we introduce in this work offers the possibility of precisely constraining the size of all the resulting shards – it is in fact not limited to constraints that produce balanced partitions. It is also worth noting that the previous cost penalty approach lacks the computational effi-

ciency of label propagation, and the largest graph that framework was originally tested on contained just 120,000 edges.

4.2.1 Partition constraints

The goal of our balanced label propagation algorithm is to take a graph $G = (V, E)$ and produce a partition $\{V_1, \ldots, V_n\}$ of V, subject to explicitly defined size constraints. The algorithm is capable of enforcing arbitrary size constraints in the form of lower bounds S_i and upper bounds T_i, such that $S_i \leq |V_i| \leq T_i$, $\forall i$. These constraints can easily take the form of balanced constraints, targeting exact balance $S_i = \lfloor |V|/n \rfloor$ and $T_i = \lceil |V|/n \rceil$, $\forall i$, or operating with leniency, $S_i = \lfloor (1-f)|V|/n \rfloor$ and $T_i = \lceil (1+f)|V|/n \rceil$, for some fraction $f > 0$.

To initialize the algorithm, we begin by randomly assigning nodes to shards, in proportions that are feasible with respect to these sizing constraints.

4.2.2 The constrained relocation problem

Given an initial feasible sharding, we wish to maintain the specified balance of nodes across shards between iterations. The key challenge is however that some shards will be more popular than others. In fact, under ordinary label propagation without any balance constraints, labelling all nodes with the same single label is a trivial equilibrium. Because we won't be able to move all nodes, our greedy approach is to sychronously move those nodes that stand to increase their colocation count (the number of graph neighbors they are co-located with) the most.

Given a constraint specification, we now formalize our greedy relocation strategy as a maximization problem subject to the above constraints. Consider therefore the nodes that are assigned to shard i but would prefer to be on shard j. Order these nodes

according to the number of additional neighbors they would be co-located with if they moved, from greatest increase to least increase, labeling them $k = 1, \ldots, K$. Let $u_{ij}(k)$ be the change in utility (co-location count) from moving the kth node from shard i to j.

Let $f_{ij}(x) = \sum_{k=1}^{x} u_{ij}(k)$ be the *relocation utility function* between shard i and j, the total utility gained from moving the leading x nodes from i to j. Observe that because $u_{ij}(k) \geq 0$ and $u_{ij}(k) \geq u_{ij}(k+1)$ for all k (since they are ordered), all $f_{ij}(x)$ are increasing and concave.

Our goal can then be formulated as a concave utility maximization problem with linear constraints.

Problem 4.2.1 (Constrained relocation)**.** *Given a graph $G = (V, E)$ with the node set partitioned into n shards $V_1, \ldots, V_n$, and size constraints $S_i \leq |V_i| \leq T_i$, $\forall i$, the constrained relocation problem is to maximize:*

$$\max_X \sum_{i,j} f_{ij}(x_{ij}) \quad \textit{s.t.} \tag{4.1}$$

$$\begin{aligned} S_i - |V_i| &\leq \sum_{j \neq i}(x_{ij} - x_{ji}) \leq T_i - |V_i|, \quad \forall i \\ 0 &\leq x_{ij} \leq P_{ij}, \quad \forall i, j. \end{aligned} \tag{4.2}$$

Here P_{ij} is the number of nodes that desire to move from shard i to j, and $f_{ij}(x)$ is the relocation utility function between shard i and j, both derivable from the graph and the partition.

For an illustration of the constraints, see Figure 4.2.

When the above problem is considered under continuous values of x_{ij}, we will now show that it reduces to a fully tractable optimization problem. Note that relaxing these integrality constraints on all x_{ij} is a fully reasonable approximation as long as the number of nodes seeking to be moved between each pair of shards is large.

We aim to rewrite the above optimization problem as a linear program. Notice that all f_{ij} are piecewise-linear concave functions. To see this, notice that the slope of f_{ij} is

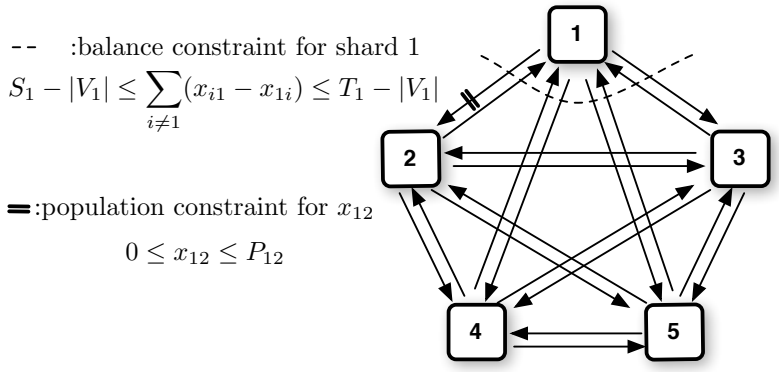

Figure 4.2: Illustration of the constraints for balanced label propagation applied to five shards. Each shard has a two-sided balance constraint, while each pair of shards has a population constraint.

constant across all intervals where the ordered users have the same derived utility. As a consequence of this, we obtain the following straight-forward lemma.

Lemma 4.2.2. *Assuming a bounded degree graph G, the objective function in Problem 1, $f(x) = \sum_{i,j} f_{ij}(x_{ij})$, is a piecewise-linear concave function, seperable in x_{ij}.*

Proof. The separability is clear from the fact that the f_{ij} depend on different variables. Since users are atomic and the graph is of bounded degree, there are a finite number of utilities, and the sum is therefore also piecewise linear. Since the nodes are sorted in order of decreasing utility, the function is concave. □

Recall that any piecewise linear concave function $f(x) : \mathbb{R}^n \to \mathbb{R}$ can be written as $f(x) = \min_{k=1,\ldots,\ell}(a_k^T x + b_k)$, for some choices of a_k's and b_k's. For our problem, all the a_k's and b_k's are scalar. Now, also recall the following [26].

Lemma 4.2.3. *Let $x \in \mathbb{R}^n$ and $f(x) = \min_{k=1,\ldots,\ell}(a_k^T x + b_k)$ be a piecewise linear concave function. Maximizing $f(x)$ subject to $Ax \leq b$ is then equivalent to:*

$$\max z \qquad \textit{s.t.} \tag{4.3}$$

$$\begin{cases} Ax & \leq & b \\ a_k^T x + b_k & \geq & z, \quad \forall k. \end{cases} \tag{4.4}$$

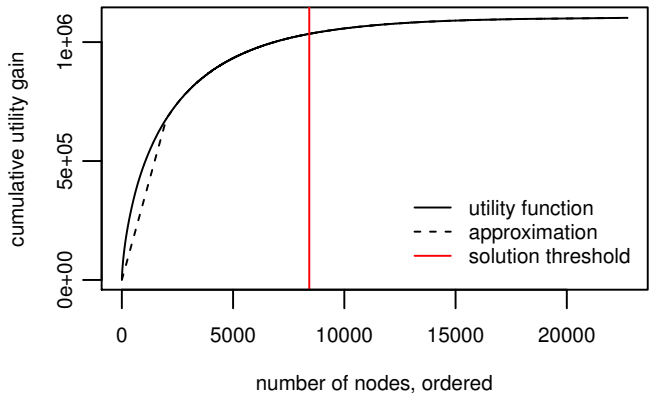

Figure 4.3: The piecewise-linear utility function for moving nodes from one shard to another, from an example problem iteration. The discontinuous derivates are imperceivable. The utilty approximation shown allows for a significant reduction in the number of constraints in the LP. The red line indicates the threshold found in the optimal solution for balanced propagation: here 8,424 of the 22,728 nodes that wanted to move were moved.

Utilizing these two lemmas, we can thus solve the concave maximization problem in Problem 1 using a linear program.

Theorem 4.2.4. *Consider a bounded degree graph $G = (V, E)$. Under continuous x_{ij}, the constrained relocation problem can be written as*

$$\max_{X,Z} \sum_{i,j} z_{ij} \quad \textit{s.t.} \tag{4.5}$$

$$\begin{aligned} S_i - |V_i| &\leq \sum_{j \neq i}(x_{ij} - x_{ji}) \leq T_i - |V_i|, \quad \forall i \\ 0 &\leq x_{ij} \leq P_{ij}, \quad \forall i, j \\ -a_{ijk} x_{ij} + z_{ij} &\leq b_{ijk}, \quad \forall i, j, k, \end{aligned} \tag{4.6}$$

where all a_{ijk} and b_{ijk} derive directly from the relocation utility functions f_{ij}. Assuming n shards and at most K unique utility gains achieved by nodes that would like to move, this constitutes a linear program with $2n(n-1)$ variables and at most $2n^2 + Kn(n-1)$ sparse constraints.

Since the most utility a node can gain is its degree, and furthermore only a small set of nodes in real world graphs have high degree, it becomes unlikely that all pairs

of shards will observe nodes seeking large utility gains. Thus, in practice the number of constraints is typically small. For $m = 78$ and $K = 100$, this would imply a linear program with 12,012 variables and 612,768 constraints, which is fully manageable by a basic LP solver owing to the extensive sparsity of the matrix of constraints.

4.2.3 Iteration

Procedurally, an iteration of this algorithm differs very little from an iteration of ordinary label propagation. First, determine where every node would prefer to move, and how much each node would gain from its preferred relocation. Second, sort the node gains for each shard pair and construct the Constrained Relocation linear program. Third, solve the linear program, which determines how many nodes should be moved, in order, between each shard pair. Fourth, move these nodes. This constitutes one iteration.

When compared to ordinary label propagation, the only difference is that rather than moving every node that asks to move, our algorithm pauses, solves a linear program, and then proceeds to move as many nodes as possible without breaking the balance. As with ordinary label propagation, the bulk of the work lies in determining where every node would prefer to move (which requires examining every edge in the graph). The entire balancing procedure has a complexity that depends principally on the number of shards and is nearly independent of the graph size (the number of constraints per shard pair, K, can depend weakly on the graph size in practice).

4.2.4 Approximating utility gain

If the number of constraints becomes limiting, we note that it is possible to greatly reduce the number of constraints by a very slight approximation of the objective function. We emphasize that constraint satisfaction is still guaranteed under this approximation,

it is merely the utility gain of the iteration that is approximated.

Observe that for each shard pair, the handful of nodes that stand to gain the most are likely to be contribute relatively unique utility levels, and so contribute many of the constraints in the problem. This is rather unnecessary, since those nodes are highly likely to move. Thus, by disregarding the unique utility levels of the first C nodes, all very likely to move, and approximating them by the mean gain of this population, we can greatly reduce the number of constraints. To exemplify this approximation, in Figure 4.3 we show one of the piecewise linear concave utility functions from a problem instance, corresponding to movement between two shards, and the threshold on the number of users that were allowed to be moved in the optimal solution.

4.3 Geographic initialization

When applying greedy algorithms to intricate objective functions, initialization can dramatically impact performance. For the balanced label propagation algorithm presented in the previous section, random initialization — in proportions that are feasible with respect to the sizing constraints — might be considered an adequate initialization. However, by using node data to initialize our assignment, we find that we can improve on both the number of iterations and the final sharding quality. How well one can do in this initial assignment depends on what auxiliary node information is available. For example, on the web one might use domain, or in a computer network one might use IP-address. In the case of the Facebook social network, we find that geographical information gives us good initial conditions. Thus, in this section we will examine an initial graph partitioning based on a geographic partitioning in the absence of any graph structure.

Facebook's geolocation services assign all 800 million users to one of approximately 750,000 cities wordwide. The idea behind our geographic partitioning is to harness the

intuitive and well-studied properties of geography as a strong basis for graph assortativity in social networks: individuals have an elevated tendency to be friends with people geographically close to them [11].

A key challenge, however, when partitioning a realtime service based on geographic information, is the heterogeneity of traffic between geographies. When balanced propagation is run under random initialization, it is our experience that the local improvements made by the algorithm tend to not discover any large-scale geographic structure. Yet some parts of the world are much more active on Facebook than others, and as a result, they have more friends and their friend-of-friend calculations are much more expensive. As a result, it is desirable to configure the geographic initialization so that shards with higher than average degree contain fewer nodes, and shards with a lower than average degree contain more.

To achieve this, instead of cutting up the geographic space to form shards of exactly equal population, we consider a more general cost model, based on the number of users in each city and also on the average degree of users in that city. Note that in this particular sharding challenge we are most concerned with sharding node attributes, and the distribution of the graph (as an edge list) is not as central a concern, but instead we are considering average degree as a means of distributing computational load. For each city c with population n_c and average degree d_c, the cost of the city is given by $\text{Cost}(c) = n_c(1 + \lambda d_c)$, where λ is a weight parameter determined by the proportion of node attribute data to edge attribute data to be sharded. For our applications we used $\lambda = 1/d_{avg}$, the reciprocal of the average degree across the graph.

4.3.1 Balloon partitioning algorithm

The partitioning algorithm we will present here constructs shards centered at the most populated cities in the data set, beginning with the most populated city not yet assigned

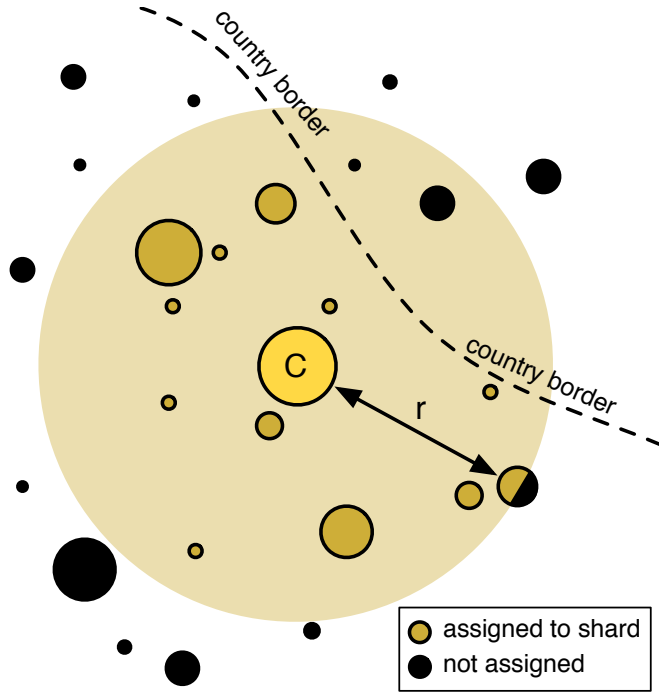

Figure 4.4: Geometric illustration of the greedy geographic initialization of a shard, with cities as circles with radii indicating cost. The algorithm centers itself at the most costly remaining city, C, and then fills a shard with the cities closest to that city, with a penalty for crossing national borders. When the shard is full, fractional assignments are made.

to a shard, and grows circular 'balloons' around these cities.

This balloon algorithm, illustrated in Figure 4.4, consists of a single iterated loop. For each of the $i = 1, \ldots, n$ shards, the goal is to obtain shards each with $(\sum_c \text{Cost}(c))/n$ in cost assigned to them. The algorithm first finds the city C with the largest unassigned cost. The city C is selected as the center-point of a new shard, and all cities with a non-zero remaining cost are sorted according to their geodesic distance from C. Since edges in the social network are overwhelmingly internal to countries, a negative distance reward is introduced to all cities in the same country as C. This ensures that the algorithm finishes assigning each country completely before moving on to another country. Beginning with C, the algorithm progresses down the sorted list of cities, and while there is capacity, it assigns each city to the shard currently being

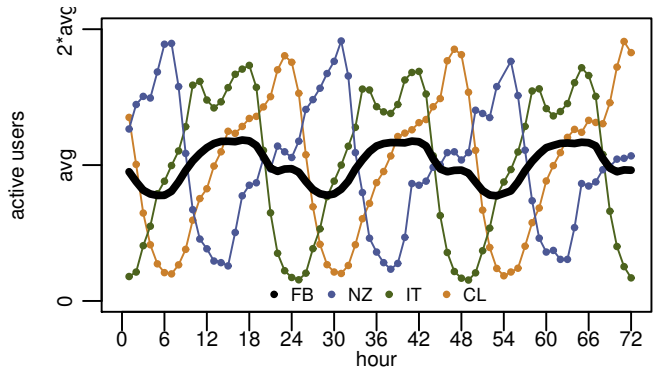

Figure 4.5: User traffic differences between countries. Comparing traffic for New Zealand, Italy, and Chile to that of Facebook as a whole, the intraday variability in users accessing the site from a single country far exceeds the variability of the site as whole.

constructed. When the algorithm does change countries, a new negative distance reward is given to cities in that country. Eventually, when there is not enough capacity in the current shard to accommodate an entire city, a fractional assignment is noted, and the cost is subtracted from the remaining cost of that city.

The result is n shards, centered over population centers, each containing all nodes within a certain radius of the central city, adaptively configured such that each shard is of equal burden under the cost model. A handful of cities are assigned to multiple shards according to a fractional division. The map of cities to shards is used to assign nodes to shards, and for those cities with distributed fractional assignments (at most n of the 750,000 cities), simple randomization is used, distributing nodes in such cities proportional to the fraction of the cost mapped to each shard. A map of the Facebook social graph partitioned into 234 partitions is shown in Figure 4.6.

After performing an initial aggregation of the social graph, this assignment algorithm operates only on the set of cities and not on the social graph itself, making it possible to quickly run this complete assignment on a single machine in a single processor thread in seconds, with no actual graph analysis being necessary.

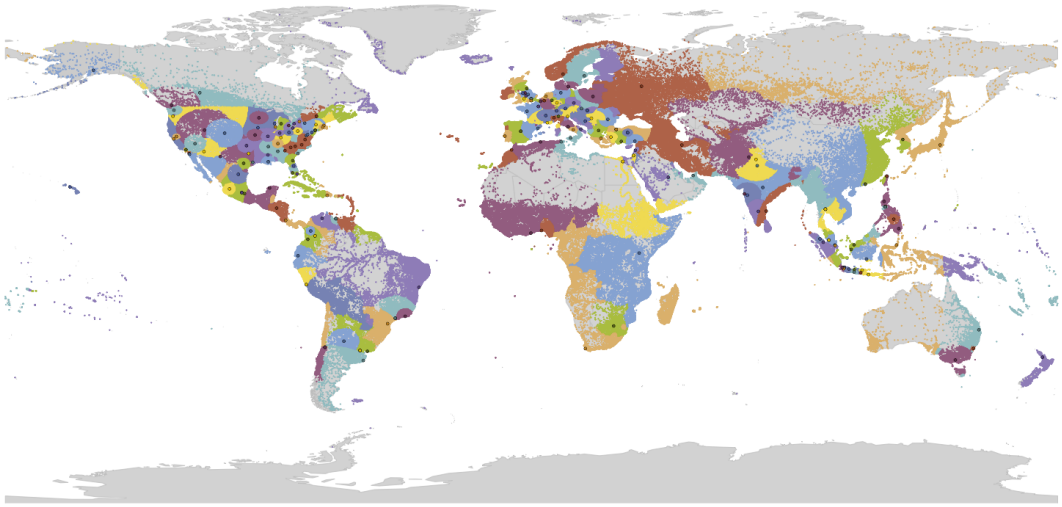

Figure 4.6: Output of the greedy geographic initialization algorithm for the $\sim 750,000$ known cities, obtaining 234 shards of equal cost, with each shard's most costly city marked. As described in the text, the algorithm is aware of national borders.

It should be noted that the geographic sharding performed here is ultimately static, and as new users register to join the site, these users can be assigned in accordance with the map from cities to shards. We note that as the geographic distribution of Facebook users slowly changes over time, re-sharding may be useful.

4.3.2 Oversharding

Selecting shards for a real-time graph computation service from a geographic initialization has another serious practical challenge that we have not yet discussed. For the purposes of load-balancing a service that handles real-time requests, it is important to mitigate potentially problematic peak loads that result from assigning geographically concentrated regions to the same shard. A three day window of user traffic is shown in Figure 4.5, where we see that local geographic regions experience much more volatile peak loads than the full site on average. Our solution to this problem was to create 3 times more shards than there are machines, and then sort shards by the longitude of their

most populated city. The shards are then distributed cyclically across the n machines so that, e.g., shard 1, $(n+1)$, and $(2n+1)$ in longitudinal order are assigned to the same machine.

4.4 Evaluating Performance

In this section we discuss the performance of our balanced label propagation algorithm as a graph cutting procedure applied to the Facebook social graph, under both random and geographic initialization. In Section 5 we evaluate shardings of this graph when deployed for a realtime graph computation service. To align the discussion between the two sections, we evaluate the algorithm by sharding the graph into 78 shards, where the service we evaluate later will consist of 78 machines, evenly split across two server racks. We also provide a comparison to performance on a publicly available social graph collected from LiveJournal [10].

4.4.1 Sharding the Facebook social graph

For the random initialization, all shards of the partition were constrained to symmetrically balanced node counts. Meanwhile, for the geographic initialization, partition size constraints were inherited from the output of the geographic balloon algorithm initialization, where partition node counts were tuned to mitigate the differences in average degree between different parts of the globe, as discussed in the previous section. All iterations were allowed a five percent leniency, $f = 0.05$, and utility approximation was used within the constrained relocation problem for the leading $5,000$ nodes between each shard pair.

The matrix of shard-shard edge counts resulting from both geographic and random initialization are shown in Figure 4.7, while the convergence properties observed when

iterating the algorithm are shown in Figure 4.8. We observe that initializing the algorithm with a greedy geographic assignment greatly accelerates the convergence of the algorithm. Using geographic initialization, before even beginning the propagation we see that 52.7% of edges are locally confined. After a single propagation, fully 71.5% of edges are local, and after four iterations this rises to 75.2%.

For the geographic initialization instance, we overshard by a factor of 3, meaning that 234 virtual shards were distributed to 78 actual shards. When oversharding for distributed computation, it is useful to distribute closely related shards across machines located on the same physical rack. While this does not effect the fraction of edges that are local to individual machines, the order in which the shards are assigned does effect the cross-rack traffic. Machines within the same rack have relatively fast high-bandwidth connections compared with machine in different racks, where all traffic must pass through a switch. Since the service we study later is split across two racks, we distribute our 234 geographic shards across the 78 machine shards by splitting them into two block groups. The first block was set to contain all shards centered in countries in North America, Africa and Oceania, and the second block contains all shards centered in South America, Europe and Asia, with shards centered in the Middle East balancing between the two sets. The two-block structure is clearly visible in Figure 4.7, but we reiterate that it does not effect the local edge fraction.

For comparison, we investigate the performance of the algorithm when initialized with a random distribution, also shown in Figure 4.8. After 8 iterations, this random initialization achieved a local fraction of 44.8%.

When nodes are initially distributed at random, it implies that a node's neighbors are initially distributed uniformly across all shards. For graphs where nodes possess many neighbors, as in the Facebook social graph, this implies that it can take many iterations until the initial symmetry of the random initialization is broken. In an important demon-

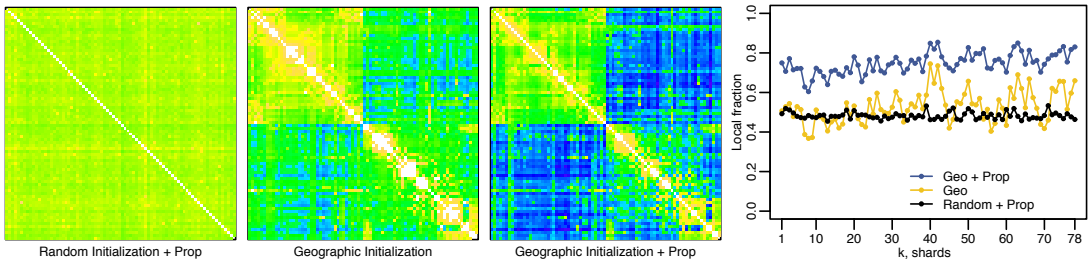

Figure 4.7: Matrices of edges between 78 shards under three different shardings: Balanced label propagation with random initialization, the sharding produced by the geographic initialization, and balanced label propagation after geographic initialization. All matrices share the same logarithmic color scale, saturated to make the structure of the random initialization visible. The fraction of local edges for each shard is also shown.

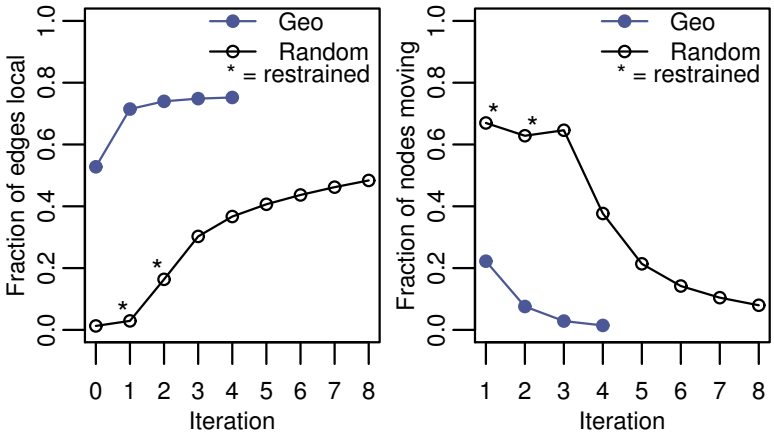

Figure 4.8: Iterating the balanced label propagation algorithm with 78 shards, from a geographic and random initialization. Left: the fraction of edges that are local as the balanced propagation is iterated. Right: the fraction of nodes that are moved in each iteration.

stration of how label propagation functions, we observed that applying our balanced label propagation algorithm from a random initial condition meant that 96.7% of nodes were relocated during the first iteration, a chaotic shuffling that slows the algorithm's ability to converge. To address this, in the random initialization shown here, the linear program constraints were modified to only move nodes that claimed to gain 2 or more additional neighbors post-propagation. This 'restrained' modification meant that only 67.0% of nodes were relocated during the first iteration. This 'restraint' was used during

the first two steps of the random initialization algorithm, as denoted in the Figure 4.8, after which it was removed and ordinary balanced label propagation was performed.

Why should one bother to move nodes that only stand to gain one additional neighbor co-location? As another instructive highlight of how balanced label propagation operates, nodes that are mostly indifferent to moving offer very useful 'slack' for the linear program: by moving them, it is possible to balance the constraints while still moving many nodes who stand to make larger gains.

The Facebook social graph is enormous, and these computations do not come cheaply. The graph aggregations required for a single iteration utilizes approximately 100 CPU days (2395 CPU hours) on Facebook's Hadoop cluster. For comparison, a simple two-sided join aggregation of the graph edge list (such as computing a shard-shard matrix in Figure 4.7) uses 72 CPU days. Once the aggregations have been performed, the linear program is solved in a matter of minutes on a single machine using lpsolve [20].

While the randomly initialized algorithm only achieves 44.8% locality compared to 75.2% locality for the geographic initialization, the random initialization produces an impressively homogenous sharding free of 'hot' shard-shard connections. It would be interesting to iterate the random initial algorithm further, but running the random initialization for 8 iteration utilized approximately 800 CPU days. Examining the properties of balanced label propagation more throughly on modest graphs would be important future work.

4.4.2 LiveJournal comparison

Because the Facebook social graph is not publicly available, we also report the performance of our algorithm when attempting to partition a large publicly available social network dataset, LiveJournal [10]. The LiveJournal graph, collected in 2006, is a

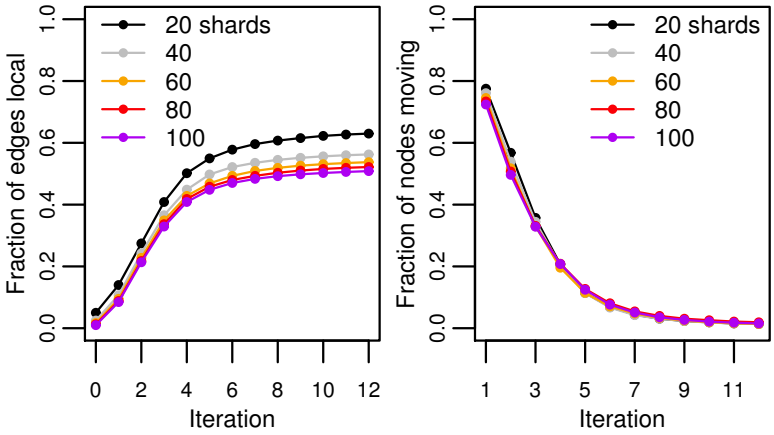

Figure 4.9: Balanced label propagation applied to the LiveJournal social graph, partitioning the graph into 20, 40, 60, 80, and 100 shards. Left: the fraction of edges that are local as the balanced propagation is iterated. Right: the fraction of nodes that are moved in each iteration.

directed graph consisting of of 4.8 million nodes and 69.0 million arcs. Because we are principally interested in partitioning undirected graphs, we consider the undirected graph consisting of 4.8 million nodes and the 42.9 million unique edges that remain and disregarding directionality.

The resulting graph is more than 20 times smaller than the Facebook graph by node count and more than 1000 times smaller by edge count. The average degree is only 8.8, making partitioning much less challenging. Because the public LiveJournal graph lacks complete geographic information, we consider only random initialization. We consider the performance of cutting the graph into 20, 40, 60, 80, and 100 symmetric shards, with five percent leniency ($f = 0.05$) and no approximation of the utility gain. Splitting into 20 shards took less than 3 minutes using a single threaded C++ implementation, while splitting into 40 shards took 8 minutes, and splitting into 100 shards took 88 minutes.

The results from partitioning the LiveJournal graph are shown in Figure 4.9, where we see that when partitioning the LiveJournal graph into 20 parts, 63% of edges are local to a single partition. Impressively, when the algorithm is scaled up to 100 shards,

fully 51% of edges are local to a single partition. Overall, we observe that the fraction of edges that are local is nearly unchanged when increasing the number of shards from 40 to 100. We interpret this performance to be a consequence of the greedy nature of the algorithm, as the algorithm principally exploits local graph relocations that are significantly below the scale of any of the shard sizes, while global improvements are less possible. A further investigation of this scaling behavior in relation to studies of natural social network cluster sizes [97] would make for interesting future work.

4.5 Realtime Deployment

In this section, we present the results of a large-scale experiment where the sharding algorithm we develop is evaluated in a realtime distributed graph computation service: Facebook's 'People You May Know' (PYMK) service for suggesting friend recommendations.

Many pages on Facebook occasionally feature a small module presenting users with 'People You May Know'. The PYMK service has contributed significantly to the growth of Facebook, accounting for around 40% of all friending on Facebook. The friend suggestions that populate this module are mostly (but not exclusively) generated by the system described in this work.

4.5.1 People You May Know

The PYMK system computes, for a given user u, and each friend-of-friend (FoF) of u, w, a feature vector $x_{u,w}$ of graph metrics based on the local structure between u and w. The system then uses machine learning to rank all the suggestions w for u, based on the graph-based feature vector $x_{u,w}$, as well as demographic features of u and w. The system described here returns the top 100 suggestions for each user (out of a potential of

many thousands of FoFs). Regeneration of suggestions is performed when a user logs in to Facebook and no recent suggestions for the user are found in the cache.

The PYMK service consists of 78 machines split across two racks (there are 40 machines to a rack, but one machine per rack is reserved as a backup). All 78 machines feature 72 GB of memory, which is used to store two in-memory indexes. First, a mirrored data structure containing basic demographic data of all 800 million users (19 GB). Second, a sharded index containing the friendlist data of those users assigned to the individual machine ($\sim$40 GB). All in-memory indexes are stored as open-addressed hash tables. To handle the full query volume, a number of identical copies of this 78 machine system run in parallel.

Prior to the optimizations presented in this work, users were assigned to machines based on their user ID mod 78, see Figure 4.1. This naive sharding has a direct advantage over any sophisticated sharding in that the shard ID is encoded directly in the user ID. Introducing the more sophisticated shardings used in this work requires adding an additional data structure serving as a *shard map*, mirrored on all machines, to map the 800 million user IDs to shard IDs.

The evaluation we perform examined three separate instances of the PYMK service operating in parallel, receiving identical and evenly balanced shares of the service load, differing only with regard to their sharding configuration. The three systems were:

- **Baseline sharding**: assigning users by the modulus sharding, 'user ID % 78'.
- **Geographic sharding**: assigning users using 234 geographic shards, with no label propagation.
- **Propagated sharding**: one step of balanced label propagation after geographic initialization.

Because of resource constraints we were only able to test three parallel systems, and did not deploy a propagated sharding featuring random initialization, which required much

more iteration and achieved worse edge localization than the unpropogated geographic initialization.

First, we characterize the impact of our sharding by evaluating our algorithm's ability to concentrate requests to few machines, reporting on the number of machines queried across requests. Next, because intelligent sharding disadvantages the PYMK service by requiring an additional round of requests (as described in the introduction), we evaluate the algorithm's ability to reduce to total query time of FoF requests, with particular attention to the slowest machine, as well as the ability to reduce the total cross-machine network traffic within the full system.

The evaluation was performed during the three day period September 20-22, 2011. Because the evaluation required the dedication of considerable hardware resources (6 racks of machines, in total 240 machines), testing on exactly identical hardware configurations was not possible (it was important to test the three systems in parallel so that no external events could impact the results). All machines featured 72GB of memory, while the machines in the baseline system and the geographic system featured 12 CPUs and the machines in the propagated system featured 24 CPUs, all 2.67 GHz Intel Xeon processors. This difference of CPU resources is one of the main reasons we focus our analysis on hardware invariant performance evaluations such as request concentration and network traffic measurements.

4.5.2 Request concentration

Here we consider the concentration of requests in a realtime setting, recording the number of machines accessed per query. Because the PYMK system has to wait for the slowest query response before performing ranking, the number of machines queried is an important performance characteristic. By waiting for fewer machines, the expected time needed to wait until the slowest machine has returned data can be significantly

decreased.

Baseline performance. In the baseline system, users are sharded by their Facebook user ID modulus 78. Prior to the sharding optimizations in this work, the distribution of requests on each machine was not instrumented, and because there were significant architectural changes to the service when the sharding optimization was introduced, the baseline system was not upgraded to include instrumentation.

Fortunately, the trailing digits of a user's ID are uncorrelated from the trailing digits of the user IDs of their friends, and thus the question of how many machines are queried during an average aggregation can be computed directly given only the degree distribution of the graph.

Consider a graph sharded across m machines. Let X_i, $i = 1, \ldots, m$ be the Bernoulli random variables indicating whether a given user has a friend on machine i. Let $Y = \sum_{i=1}^{m} X_i$ be the total number of machines that are queried when this user's friends data is aggregated. By the law of total probability,

$$Pr(Y = k) = \sum_{d} Pr(Y = k | \deg(u) = d) \cdot Pr(\deg(u) = d),$$

where $\deg(u)$ is the degree of the user, and $Pr(\deg(u) = d)$ is the empirical Facebook degree distribution. Focusing on the term $Pr(Y = k | \deg(u) = d)$,

$$Pr(Y = k | \deg(u) = d) = Pr\left(\sum_{i=1}^{m} X_i = k | \deg(u) = d\right),$$

where X_i are Bernoulli distributed random variables. To derive the distribution of X_i, notice that, $\forall i$,

$$\begin{aligned} Pr(X_i = 1 | \deg(u) = d) &= \\ &= 1 - Pr(X_i = 0 | \deg(u) = d) \\ &= 1 - (1 - 1/m)^d. \end{aligned}$$

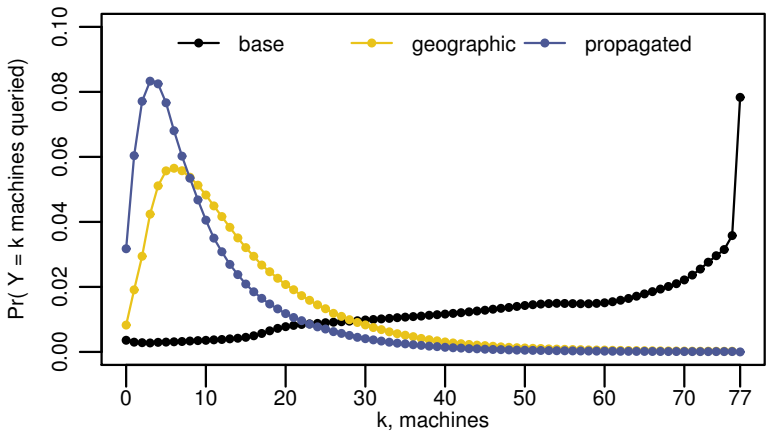

Figure 4.10: Number of non-local machines queried per request during friend-of-friend calculations in PYMK. The median number of machines queried for the baseline, geographic, and once-propagated shardings were 59, 12, and 9 machines, respectively.

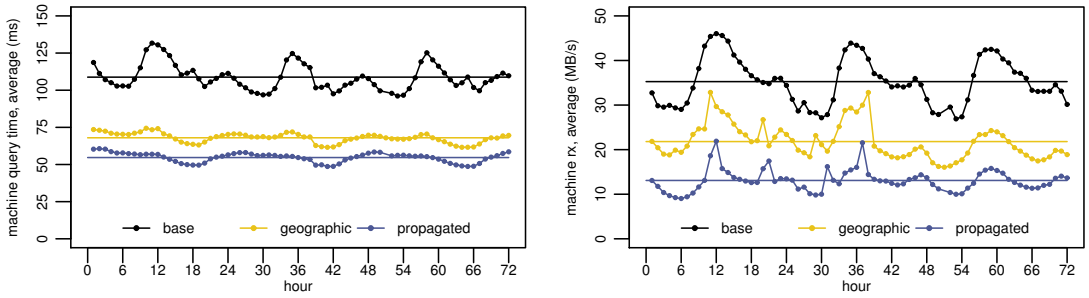

Figure 4.11: Query time and network traffic for the three different shardings applied to the PYMK service. Because network traffic is instrumented on a machine level, the data also captures daily traffic bursts which correspond to loading data into the service. For this reason, momentary outliers should be considered benign.

Thus, we see that $X_i \sim$ Bernoulli$(1 - (1 - 1/m)^d)$, $\forall i$. While these m variables are identically distributed, they are unfortunately not independent.

Since the variables X_i are not independent, we resort to a simple Monte Carlo simulation of the distribution $Pr(Y = k | \deg(u) = d)$ for a uniformly sharded system. Combing this distribution with the empirical degree distribution from PYMK queries, $Pr(\deg(u) = d)$, gives us the theoretical request concentration distribution for the baseline system.

Results. In Figure 4.10, we compare the distribution of requests as measured for the two algorithms and computed for the baseline system. Note that our measurement of request concentration is not affected by differences in hardware or traffic volumes between the three systems.

The median number of non-local machines queried for the baseline, geographic, and once-propagated shardings were 59, 12, and 9 machines, respectively. Notice that under the old naive system, the most common occurrence was that friend lists had to be aggregated from all 77 non-local machines, while the modes for the new shardings are 6 non-local machines for the geographic sharding and 3 non-local machines for the once-propagated sharding.

4.5.3 Query time and network traffic

Here we report on the relative performance of the three systems with regard to query time and network traffic. While the hardware specifications of the three PYMK services were not identical, with the propagated sharding operating with twice as many CPUs per machine, we still report the query times, noting that each query was run in a single thread, and that for the most part, the number of cores per machine did not come into play. The baseline and geographic systems were run on identical hardware, and in any event the bandwidth comparisons are independent of the machine specifications and can thus be taken at face value.

The time-averages of the average machine query times for the baseline, geographic, and once-propagated shardings were 109ms, 68ms, and 55ms, respectively. The time-averages of the maximum machine query times were 122ms, 106ms, and 100ms, respectively (not plotted). Notice that the geographic system, which operated on hardware identical to the baseline system, featured an average query time only 62.3% of the baseline system. The total improvement when comparing to the propagated system was an

average query time only 50.5% of the baseline system.

Recall that the two optimized systems being tested are disadvantaged compared to the baseline system because they must perform an additional query redirection, since the web tier does not possess a copy of the shard map and doesn't know which of the 78 machines the user lives on.

Meanwhile, when we turn to network traffic, the time-averages of the average machine traffic for the baseline, geographic, and once-propagated shardings were 35.3 MB/s, 21.8 MB/s, and 13.1 MB/s, respectively. The time-averages of the maximum machine traffic (not plotted) were 103.6 MB/s, 68.0 MB/s, and 51.0 MB/s, respectively. The systems were configured to be equally load-balanced, each handling equal thirds of the total traffic. Thus, the machines in the propagated system saw network traffic levels only 37.1% of the baseline system machines.

4.6 Discussion

The problem of clustering a graph for community detection is a widely studied active area of research within computer science and physics [115, 119]. In this work, we approach the rather different challenge of graph partitioning. We develop and evaluate a novel algorithm, balanced label propagation, for partitioning a graph while managing these challenges.

We show that by using intelligent partitioning in the context of load-balancing a realtime graph computation service, we are able to dramatically outperform a baseline configuration. While a random initialization of our balanced label propagation algorithm produces an impressive sharding, we show that by using user metadata we can derive a sharding that is greatly superior to a random initialization while still maintaining uniformity in shard sizes. These techniques were applied to a single system in this work, but we believe that they are broadly applicable to any graph computation system

that distributes graphs across many machines.

CHAPTER 5

RESTREAMING GRAPH PARTITIONING: SIMPLE VERSATILE ALGORITHMS FOR ADVANCED BALANCING

Partitioning large graphs is difficult, especially when performed in the limited models of computation afforded to modern large scale computing systems. In this work we introduce *restreaming graph partitioning* and develop algorithms that scale similarly to streaming partitioning algorithms yet empirically perform as well as fully offline algorithms. In streaming partitioning, graphs are partitioned serially in a single pass. Restreaming partitioning is motivated by scenarios where approximately the same dataset is routinely streamed, making it possible to transform streaming partitioning algorithms into an iterative procedure.

This combination of simplicity and powerful performance allows restreaming algorithms to be easily adapted to efficiently tackle more challenging partitioning objectives. In particular, we consider the problem of *stratified graph partitioning*, where each of many node attribute strata are balanced simultaneously. As such, stratified partitioning is well suited for the study of network effects on social networks, where it is desirable to isolate disjoint dense subgraphs with representative user demographics. To demonstrate, we partition a large social network such that each partition exhibits the same degree distribution in the original graph — a novel achievement for non-regular graphs.

As part of our results, we also observe a fundamental difference in the ease with which social graphs are partitioned when compared to web graphs. Namely, the modular structure of web graphs appears to motivate full offline optimization, whereas the locally dense structure of social graphs precludes significant gains from global manipulations.

5.1 Introduction

The tremendous scale of modern graph datasets has rapidly increased the demand for efficient algorithms for graph analysis. With the World Wide Web featuring over a trillion URLs and online social networks such as Facebook featuring more than a billion active users, it is becoming increasingly difficult to perform even the simplest graph computations.

The tractability of large-scale graph computations often hinges upon the ability to efficiently partition a graph for distributed computation. The scale of this partitioning varies depending on the domain, but the lesson is the same: partitioning massive graphs for distributed computation can greatly decrease both network communication and runtime (see Chapter 4), while even in-memory computations can benefit from partitioned graph arrangements [92].

But partitioning large graphs is difficult, especially within modern limited models of large-scale computation. Responding to this, the goal of *streaming graph partitioning* is to partition the node set of a graph into k balanced disjoint subsets by serially examining only individual nodes and their local adjacency list. Importantly, a streaming graph partition algorithm is forced to make a permanent partition assignment the very first (and only) time it examines each node, as opposed to allowing the partitioning to come from post-processing, as in the semi-streaming model of computation [4]. The motivation for streaming graph partitioning is that often times the distributed systems performing graph computations are 'anyways' required to load a graph from a datastore, and one might as well execute this loading process – streaming the graph to the computation system – in an intelligent manner.

We introduce *restreaming graph partitioning*, which is motivated by situations where the same graph – or approximately the same graph – can be expected to be repeatedly streamed on a regular basis. After all, if a graph is going to be reloaded from

a datastore with any regularity, streaming graph partitioning is making it unnecessarily difficult for itself by starting over from scratch with each stream. Instead, restreaming graph partitioning retains node assignments across streams, allowing subsequent streams to produce partitionings with fewer cut edges.

In fact restreaming can produce partitions of such quality and with such modest memory requirement that restreaming graph partitioning merits serious consideration as an efficient iterative streaming algorithm well outside the motivating 'data loading' context. Surprisingly, we find that after only a handful of restreams, our restreaming graph partition algorithms converge upon graph partitions competitive with or even superior to a fully offline partitioning algorithms, METIS [79], in a number of important instances.

In particular, restreaming graph partition algorithms cut fewer edges than METIS in social graphs, though they cut more edges in web graphs. Indeed, it is well understood that social graphs and web graphs are quite different in structure [151, 37]. We posit that there is also a fundamental difference in the partitioning of web and social graphs. While the local dense structure of social graphs precludes very high quality partitionings, it rewards the local greedy moves of restreaming graph partitioning. Meanwhile, the fully offline optimization of METIS is able to discover the extremely high quality partitions of web graphs through multi-level coarsening and non-greedy Kernighan-Lin refinement [83]. Given the increasing size and importance of social graphs, it is important to develop new lightweight algorithms specially designed with these in mind.

Towards this goal, we construct restreaming versions of the streaming partitioning algorithms *Linear Deterministic Greedy* (LDG) and *FENNEL*, algorithms developed in [142] and [149, 150] respectively, both greedy heuristic approaches to partitioning. Where Linear Deterministic Greedy uses multiplicative weights to guarantee balance, FENNEL mimics modularity maximization [19, 120] by using regularization to direct a greedy assignment strategy towards balance. This regularization approach does not it-

self guarantee balanced partitions, and as part of this work, we show how one can 'temper' such a regularization over the course of restreams to obtain a restreaming variation on FENNEL that can ultimately guarantee balance in a way that ordinary modularity maximization can not.

Given that restreaming these highly scalable algorithms can bring them into competition with fully offline methods, we show how their scalability makes it possible to adapt towards much more sophisticated objectives. Indeed, for certain types of distributed graph computation it can be desirable to obtain more sophisticated notions of balance than just the number of nodes [79]. It is straight forward to modify any of the streaming and restreaming algorithms we consider to balance any cumulative node attribute, for example the total degree of each partition, (or as in [149], the number of internal edges on each partition).

But beyond simply balancing one attribute, we show that significantly more sophisticated notions of balance, similar to *multi-constraint* balance from high-performance computing [79], are obtainable. First, we show how the multiplicative weights in LDG can be modified to balance both node count and edge count at once. Moreover, we show how restreaming LDG and FENNEL can be adapted to efficiently perform *stratified graph partitioning*, a constrained graph partitioning problem we introduce that aims not just to balance nodes across partitions, but also ensure that each partition of the graph exhibits a balanced proportion of nodes from an arbitrary number of strata. This offers an important contribution for the study of social networks, making it possible to create dense balanced clusters, where each cluster contains an equal proportion of users from several age strata, countries, activity levels, and friend counts. As an important demonstration, we study *degree-stratified graph partitioning*, where each balanced cluster is required to exhibit the same degree distribution.

The social network example above addresses an important problem in online so-

cial network experimentation [53]. In such experiments, one wishes to select treatment and control groups that are structurally isolated from each other in order to minimize spillover effects. Without stratified balance constraints, it is natural to partition a social graph either geographically or according to some other basis of assortativity. As a result, ordinary graph partitioning, without stratified balance, risks producing graph partitions that are highly heterogenous, none of the partitions being representative. In introducing stratified graph partitioning, we hope to contribute a highly scalable partitioning methodology useful as a stratification technique for variance reduction in network experimentation (see Chapter 6), and also cross validation on graphs [116].

Lastly, we discuss parallelization. A notable drawback of single-shot streaming partitioning is that it is fundamentally serial, making parallelization difficult without continuous communication between parallel workers [149]. These algorithms are specifically intended for partitioning extremely large graphs, and we show how restreamed graph partitioning can be easily parallelized – communicating only between stream iterations – at only a small cost in the final partition quality.

5.2 Streaming partitioning

Multi-way graph partitioning is a classical NP-hard problem. Even the two-way partitioning problem *minimum graph bisection* is NP-hard [59], with the best known polynomial time approximation algorithm achieving only a $O(\sqrt{n}\log(n))$-factor approximation [52] for general graphs. Similarly, semi-streaming algorithms can guarantee weak approximation bounds for graph cuts while utilizing only $O(n\text{polylog}(n))$ memory [4]. Meanwhile, a robust community of research has emerged to develop efficient algorithms that achieve good performance on real world graphs. Among existing offline algorithms, we focus on the METIS package [79] for graph partitioning, and use METIS as our running basis for comparison when comparing online to offline methods.

In this section, we review streaming graph partitioning and introduce restreaming graph partitioning. We show how FENNEL, a streaming algorithm previously without balance guarantees, can be 'tempered' to guarantee balance. We then discuss how our restreaming framework is capable of both managing dynamic graph partitioning and efficient parallelization.

5.2.1 The streaming model

We now review the basic details of the streaming partitioning model. Let $P^t = \{P_1^t, \ldots, P_k^t\}$ denote a k-way partitioning of the node set at time t, where P_i^t is the set of nodes in partition i at time t and $P^t(u)$ denotes the partition that contains node u. A streaming algorithm is sequentially presented a node u and its neighbors $N(u)$, and it must assign u to a partition i utilizing no more information than contained in the current partitioning P^t. Over the course of a stream, the time counter advances by one for each node it examines.

Since streaming graph partition algorithms make decisions based on an incomplete but increasing amount of information, the order in which data is streamed can affect performance, and worst-case orders can easily undermine the streaming approach [142, 149]. However, it has generally been observed that presenting the data in either a breadth first, depth first, or in a random order does not greatly alter performance [142, 149]. Of these orders, a random ordering is the simplest to guarantee in large-scale streaming data scenarios, and so we restrict our analysis to only consider random node orders for simplicity. When considering restreams later on, we focus on persistent random orders.

Finding partitions that are strictly balanced, where $|P_i| = |P_j|$ for all i and j, is rarely necessary. As a result, many partitioning algorithms [80, 149] include a 'slackness' parameter, explicitly or implicitly allowing deviations from exact balance, often in exchange for superior cuts. As part of this work, we present algorithms for exact

balance and also modifications for 'slacked' balance.

Stanton and Kliot [142] considered a broad range of heuristics for performing streaming node assignment. Of these heuristics, the method with the best performance was 'Linear Deterministic Greedy' (LDG), where each node u is assigned to the partition

$$\underset{i\in\{1,\ldots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| \left(1 - \frac{|P_i^t|}{C_i}\right), \tag{5.1}$$

where C_i is the maximum capacity of partition i. Notice that since this examines each node but once, $|P_i^t \cap N(u)|$ will be exactly 0 for many nodes at the start of the stream, and $|P_i^t \cap N(u)|$ is only likely to reflect the actual number of neighbors a node shares with a partition near the end of the stream. Single shot LDG exhibits impressive performance despite this handicap. While the original investigation of LDG was merely heuristic, subsequent work has shown that an algorithm inspired by LDG is capable of recovering a planted partitioning from a basic infinite random graph model, and also that no streaming algorithm can obtain an $o(n)$ approximation with a worst-case or random stream ordering on an arbitrary graph [143].

Meanwhile, FENNEL [149], a streaming generalization of modularity maximization, attempts to maximize the following objective function:

$$H = \sum_{u\in V} |P^t(u) \cap N(u)| - \frac{\alpha}{2} \sum_{i=1}^{k} |P_i^t|^\gamma. \tag{5.2}$$

Notice that when $\gamma = 2$, the regularization becomes functionally equivalent to $\alpha \sum_i \binom{|P_i^t|}{2}$, which is equivalent to modularity maximization with an Erdős-Rényi baseline with probability α. As a streaming greedy maximization, maximizing this objective function corresponds to assigning u to the partition that maximizes the change $\Delta H_i^t(u) = |P_i^t \cap N(u)| - \frac{\alpha}{2}[(|P_i^t| + 1)^\gamma - (|P_i^t|)^\gamma]$. To first order, this corresponds to maximizing $|P_i^t \cap N(u)| - \alpha\frac{\gamma}{2}(|P_i^t|)^{\gamma-1}$, where the first order approximation is exact

for $\gamma = 2$. Thus the FENNEL assignment rule is:

$$\underset{i \in \{1,\ldots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| - \alpha \frac{\gamma}{2} (|P_i^t|)^{\gamma - 1}. \tag{5.3}$$

In this work we focus our analysis of FENNEL on the special case of $\gamma = 2$, namely streaming modularity maximization.

While the multiplicative weights of LDG enforce exact balance, the additive regularization used by FENNEL only ensures approximate balance. While it is straightforward to show that this assignment mechanism must produce exact balance for $\alpha > \lceil \frac{n}{k} \rceil$, such a large α focuses almost entirely on balancing and leads to a very poor partitioning. Nonetheless, when run at appropriately chosen values of α, FENNEL performs very well on a number of real world networks, producing very nearly balanced partitions [149].

The first phase of many common multiphase modularity maximization algorithms, including the Louvain method [19] and modularity-specialized label propagation [102], bear a clear similarity to FENNEL. The connection between regularizing label propagation and modularity maximization was also outlined by Barber and Clark [18]. By restreaming FENNEL, we show how modularity maximization also fits well within a restreaming framework.

5.3 Restreaming Partitioning

For the distribution of very large graphs, the utility of streaming graph partitioning derives from the routine need to stream graph datasets, and when performing this streaming it can be worthwhile to attempt to partition the dataset with some intelligent assignment mechanism. It is equally routine, however, that the streaming process is repeated periodically, and often frequently.

For example, a social networking service might be interested in a streaming parti-

tioning algorithm because it loads a graph from memory to dedicated ranking servers on a daily basis (see Chapter 4). However, if a streaming algorithm sees nearly the same data routinely, it is clearly worth considering what information can be retained between streams so as to improve performance.

We thus introduce the concept of restreaming graph partitioning, and in particular we present restreaming versions of LDG and FENNEL, the two single-shot streaming graph partitioning algorithms presented earlier. In our restreaming framework, subsequent streams of LDG and FENNEL have access to the result of previous streams. We consider a graph as being streamed in a random but persistent order each time it is restreamed, and we use persistent (de-randomized) tie breaking across restreams.

5.3.1 Restreaming LDG

In the case of restreaming LDG, P_i^t records the most recent partition assignment, either from the previous stream or, when present, from the current stream. Additionally, let x_i^t record the number of nodes assigned to i during the current stream. The assignment rule for restreaming LDG remains functionally similar to (5.1),

$$\underset{i\in\{1,\dots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| \left(1 - \frac{x_i^t}{C_i}\right). \tag{5.4}$$

Since each x_i^t increases over each stream from 0 to C_i, the partitioning achieves exact balance at the end of each stream. Conceptually, restreaming LDG resembles a repeated shooting method, where each time the partitions are built up anew, with the benefit of the probable assignments for nodes not yet seen in the current stream. Since LDG matches the constraints C_i after each restream, it is ideal for applications with hard constraints, otherwise these hard constraints can be loosened by setting $\Sigma_i C_i > n$.

5.3.2 Restreaming FENNEL

FENNEL can be restreamed without any change to its objective function. Whereas restreaming LDG rebuilds the partitioning each time, and thus involves implicit notions of the beginning and end of a stream, FENNEL's objective function can be computed without knowing its location in the stream. On the other hand, this property of FENNEL prevents it from reaching exact balance after a single stream. In the restreaming scenario, however, we show that it is possible to achieve exact balance using FENNEL by 'tempering' the solution towards increasingly balanced partitions over repeated restreams. Namely, with each restream we run FENNEL with a larger value of parameter α, denoting the value of α during stream s as α_s. In this way tempering increasingly emphasizes balance, while granting time in the earlier streams to finding high quality partitions. Alternatively, had FENNEL been run with too high an initial α, the algorithm would have resorted to placing nodes in partitions based solely on balance and without regard to the quality of the partitions. As noted earlier, once each node is reconsidered by a stream of FENNEL for which $\alpha_s > \lceil \frac{n}{k} \rceil$, the assignment mechanism will necessarily return a balanced partition. We formalize this observation through the following proposition.

Proposition 5.3.1. *If $\alpha_s > \lceil \frac{n}{k} \rceil$ then at the completion of restream s, $|P_i^t| \in \{\lfloor \frac{n}{k} \rfloor, \lceil \frac{n}{k} \rceil\}$ for all i.*

Proof. Suppose not: then at some time $\tau \leq t$ a node u was assigned to a partition i where $|P_i^\tau| - |P_j^\tau| \geq 1$ for j being the smallest partition. Since $|P_i^\tau \cap N(u)| \leq |P_i^\tau|$:

$$\begin{aligned}
\Delta H_i^\tau(u) &= |P_i^\tau \cap N(u)| - \alpha_s |P_i^\tau| \\
&\leq |P_i^\tau| - \alpha_s |P_i^\tau| \\
&= -|P_i^\tau|(\alpha_s - 1), \\
\Delta H_j^\tau(u) &= |P_j^\tau \cap N(u)| - \alpha_s |P_j^\tau| \geq 0 - \alpha_s |P_j^\tau|.
\end{aligned}$$

Then:

$$\begin{aligned}
\Delta H_i^\tau(u) - \Delta H_j^\tau(u) &= \alpha_s |P_j^\tau| - |P_i^\tau|(\alpha_s - 1) \\
&\leq \alpha_s |P_j^\tau| - (|P_j^\tau| + 1)(\alpha_s - 1) \\
&= |P_j^\tau| + 1 - \alpha_s.
\end{aligned}$$

As P_j^τ is the smallest partition, $|P_j^\tau| \leq \lceil \frac{n}{k} \rceil - 1$, meaning that $\Delta H_i^\tau(u) - \Delta H_j^\tau(u) < 0$ — a contradiction as u would have then been assigned to partition j. □

When the maximum degree $d < \lceil \frac{n}{k} \rceil$ it can be shown the requirement is relaxed to $\alpha_s > d$. It's also clear that if restreaming FENNEL finds a balanced partition for some $\alpha < \lceil \frac{n}{k} \rceil$ then further tempering will not change that partition.

Proposition 5.3.2. *If at some time t_0, $P^{t_0+j} = P^{t_0}$ for all $j = 1, \ldots, n$ (one complete stream), $|P_i^{t_0}| \in \{\lfloor \frac{n}{k} \rfloor, \lceil \frac{n}{k} \rceil\}$ for all i, and $\alpha_{s+1} \geq \alpha_s$ for all s then $P^t = P^{t_0}$ for all $t > t_0$.*

Proof. We prove this by induction. Suppose no node has moved from time t_0 to some $t > t_0 + n$, and node u in partition j is the next node in the stream at time $t + 1$. Since no nodes have moved for time n, $|P_i^{t+1-n} \cap N(u)| = |P_i^{t+1} \cap N(u)|$ and thus:

$$\begin{aligned}
\Delta H_j^{t+1} - \Delta H_j^{t+1-n} &= -(\alpha_{s+1} - \alpha_s)(|P_j^{t+1}| - 1), \\
\Delta H_i^{t+1} - \Delta H_i^{t+1-n} &= -(\alpha_{s+1} - \alpha_s)|P_i^{t+1}| \quad u \notin i.
\end{aligned}$$

Since α is increasing and $|P_j^{t+1} - 1| \leq |P_i^{t+1}|$ then $\Delta H_j^{t+1} - \Delta H_j^{t+1-n} \geq \Delta H_i^{t+1} - \Delta H_i^{t+1-n}$. Thus, as u was assigned to j at time $t+1-n$, and ties are broken consistently, u will also be assigned to j at time $t+1$. □

As we discuss later, when using tempered FENNEL to partition real world graphs we observe that the quality of the final partitioning is relatively insensitive to the initial value of α_0. This is a somewhat surprising observation that has the added benefit of removing the α_0 selection problem present in single stream FENNEL. Meanwhile, there remains a trade off between computation and performance in choosing how fast to temper, though our empirical results suggest that moderate numbers of restreams are typically sufficient.

5.3.3 Convergence over restreams

Every node allocation/relocation in FENNEL increases its objective function. As there are only a finite number of different possible partitions, FENNEL will converge to a final partitioning at any fixed α given enough restreams, even if α is not tempered all the way to the bound established in Proposition 1. But since α is in theory a continuous parameter, we may be concerned that the solutions differ at exponentially different values of α. While it is not of practical importance — since it is always possible to make large changes in α when tempering — we establish a resolution limit of α, the granularity below which the partitioning solution can not change. We show that there are only polynomially many unique values of α for which any changes in partitioning can occur. We emphasize that this resolution limit is much finer than the amount that we choose to increase α by in practice, but this investigation illustrates important structures of the tempering framework.

Proposition 5.3.3. *For some $\alpha_s > 0$, and a partitioning P^t, then for any increasing sequence of L values $\alpha_s < \alpha_{s+1} < \ldots < \alpha_{s+L} \leq \alpha_s + \frac{1}{n^2}$ on which FENNEL is*

repeatedly restreamed to convergence, there is at most one value of $\alpha_{s+\ell}$, $0 \leq \ell \leq L$, *such that the converged partitioning at* $\alpha_{s+\ell}$ *is different from the converged partitioning at* $\alpha_{s+\ell+1}$.

Proof. Suppose there are two distinct pivotal $\alpha_{s+\ell}$, denoted, α_a and $\alpha_b > \alpha_a$, such that at α_{a+1} and α_{b+1} FENNEL converges to a different partition than at α_a and α_b. Let $\delta_{ij}^a(u) = |P_i^{t_a} \cap N(u)| - |P_j^{t_a} \cap N(u)|$, and $x_{ij}^a = |P_i^{t_a}| - |P_j^{t_a}|$. It must then be that there are partitions i, j, p and q and nodes u and v such that for these different α values, nodes would switch between partitions implying a change in sign of $\Delta H_i^{t_a}(u) - \Delta H_j^{t_a}(u)$ and of $\Delta H_p^{t_b}(v) - \Delta H_q^{t_b}(v)$ giving:

$$\delta_{ij}^a(u) - \alpha_a x_{ij}^a \geq 0 \qquad \delta_{ij}^a(u) - \alpha_{a+1} x_{ij}^a < 0$$
$$\delta_{pq}^b(v) - \alpha_b x_{pq}^b \geq 0 \qquad \delta_{pq}^b(v) - \alpha_{b+1} x_{pq}^b < 0.$$

Notice that it must be that $x_{ij}^a \neq 0$ and $x_{pq}^b \neq 0$. Without loss of generality assume that each $x_{ij}^a > 0$ and $x_{pq}^b > 0$, then for u or v to be assigned to i and p respectively, it must be that $\delta_{ij}^a(u) \geq 0$ and $\delta_{pq}^b(v) \geq 0$ as well. Thus it can be shown that: $(\alpha_{b+1} - \alpha_a) x_{ij}^a x_{pq}^b > \delta_{pq}^b(v) x_{ij}^a - \delta_{ij}^a(u) x_{pq}^b > 0$. Notice then that $c = \delta_{pq}^b(v) x_{ij}^a - \delta_{ij}^a(u) x_{pq}^b$ is both positive and an integer, yielding that $\alpha_{b+1} - \alpha_a > \frac{c}{x_{ij}^a x_{pq}^b} > \frac{1}{n^2}$, a contradiction. □

Note that this convergence is in theory incredibly slow. In practice it is vastly more efficient to increase α in larger steps, and without waiting for convergence at each value of α. Indeed, the tempering results we present later correspond to increasing α at an exponential rate, from an initial α_0 to the critical α for which FENNEL is guaranteed to be balanced.

When restreaming FENNEL, tempered or untempered, it ultimately converges only to one of many local maxima of its modularity-like objective function. As discussed in [62], modularity typically has many high quality local maxima, which is of great practi-

cality if one merely needs to find high quality partitions, but also of grave concern when using modularity to discern 'community structure', something we are not attempting.

LDG does not have any of the same convergence guarantees. Indeed, restreaming LDG does not necessarily converge. Furthermore, should it converge, the resulting partitioning would depend upon the specific node ordering: if the graph was restreamed in a different order then nodes would be very likely to move. By comparison, the convergence of FENNEL and tempered FENNEL outlined above do not depend on any persistence in the node order. Despite the lack of convergence guarantees, LDG performs well, returning a balanced set after each restream. This lack of convergence guarantee is also one of LDG's strengths, enabling it to handle dynamic graphs very well.

5.3.4 Dynamic graphs

In real world settings, large empirical graph datasets are typically not static graphs, but rather they are slowly varying in time, with their edges sets evolving gradually relative to their immense size. In such cases, the graph may be expected to change slightly between streams, and it is important to consider the ability of both restreaming algorithms to accommodate dynamic graphs. One advantage of restreaming LDG is that it doesn't require any modification: since LDG rebuilds the graph each stream there aren't any restrictions on how the graph changes each time.

On the other hand, restreaming FENNEL is able to accommodate dynamic graphs only when α is held fixed in a manner similar to ordinary single stream FENNEL. When FENNEL is tempered across restreams, the resulting partitioning becomes increasingly rigid, unable to adjust to dynamic changes in a graph. In this way, FENNEL is only appropriate for dynamic graphs in its untempered form, precluding situations where exact balance is important.

5.3.5 Parallelization

Despite the simple computations and manageable memory footprint involved in LDG and FENNEL, in some settings the sheer size of the dataset being streamed may make parallelization highly desirable. Indeed, parallelizing single stream LDG and FENNEL is possible, but requires that a list of size $O(n)$ on each parallel thread is kept concurrent. In contrast, restreaming LDG and FENNEL can be parallelized without any communication during a stream, instead relying purely on inter-stream communication; thus speeding the streaming process by a factor equal to the number of machines used. Namely, for W workers, each of which will see a unique random $\frac{1}{W}$ fraction of a stream, we parallelize restreaming LDG and FENNEL in the following way. Each worker partitions their own $\frac{n}{W}$ nodes by utilizing the previous streams partitioning for nodes not in their stream, and the most recent destination of those $\frac{n}{W}$ nodes in their stream. Between restreams, each worker reports on their share of the partitioning and this compiled list is distributed to all workers for the next restream. For the first stream, we can initialize the partitioning utilizing a hash function applied to the node indices. Notice that this puts the first stream of LDG and FENNEL at significant performance disadvantage, but interestingly, this is largely overcome by additional restream iterations.

Thus these algorithms can be parallelized without communication for only a small partition quality tradeoff. Note that the parallelized implementation of the offline partitioning package METIS [80] also requires a similarly small quality tradeoff.

5.4 Generalized types of balance

In many situations the true objective function may depend not on balancing nodes, but on balancing edges, a combination of nodes and edges or some other more complicated function. In this section, we show how our restreaming algorithms can be modified

to guarantee more general types of balance. In particular, we present a new balancing objective we call *stratified graph partitioning*, where an arbitrary number of node strata are each required to be balanced.

5.4.1 Balancing other quantities

The simplicity and directness of LDG and modularity maximization allow for straightforward generalizations. Indeed, notice that we can modify restreaming LDG's objective function, Equation 5.4, to balance the sum of the degrees of each partition. The objective function can be modified as:

$$\underset{i\in\{1,\ldots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| \left(1 - \frac{x_i^t}{C_i}\right), \tag{5.5}$$

where $x_i^t = \Sigma_{u\in P_i^t}|N(u)|$ is the sum of the degrees in P_i^t and C_i is set to be $\lceil \frac{m}{k} \rceil$. More generally, x_i^t can be the sum of any positive node weights c_u, and each $C_i = \frac{1}{k}\sum_u c_u$, is simply the total possible sum split k ways. In this framework $c_u = 1$ corresponds to node balance, $c_u = |N(u)|$ corresponds to balancing degrees, and $c_u = 1 + \frac{n}{2m}|N(u)|$ treats node balance and degree balance as equally important. Here c_u can in fact be any arbitrary positive attribute calculated for each u a priori, such as the number of friends of friends on a social network, or the number of log records each node produced in the past month.

Note that when running this more general version of LDG, exact balance of the node attribute is no longer precisely guaranteed due to granularity. For example, when balancing degree and assigning a very high degree node late in a stream, that node will invariably push the sum term x_i^t in Equation 5.5 over the threshold C_i. Thus LDG will only balance partitions to within the maximum value of c_u.

Similarly to LDG, FENNEL can be reinterpreted as a function on the total weights of each partition, rather than just the number of nodes. The general assignment rule can

then be stated simply as

$$\underset{i\in\{1,\ldots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| - \alpha x_i^t. \tag{5.6}$$

The above modifications make it possible to address alternative notions of balance, which raises the important question of whether multiple balancing objectives can be obtained simultaneously. For LDG, we note that it is possible to adjust the multiplicative weights to attempt to balance multiple objectives simultaneously. Consider adjusting LDG such that u is assigned to:

$$\underset{i\in\{1,\ldots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| \left(1 - \frac{x_i^t}{C_i}\right) \prod_{\ell} f_\ell((\bar{y}_{\ell,i}^t - \bar{y}_\ell)(\bar{y}_\ell - y_{\ell,u})) \tag{5.7}$$

where $\bar{y}_\ell$ is the average value of objective ℓ on G, $\bar{y}_{\ell,i}^t$ is the average value of ℓ in P_i^t at time t, $y_{\ell,u}$ is the value of ℓ at u and $f_\ell(x)$ are positive increasing functions. For example, if the argument of f_ℓ is $(d_i - \frac{2m}{n})(\frac{2m}{n} - d_u)$ where d_i is the average degree of nodes in P_i^t, and d_u is the degree of u, then notice that the quantity is positive if and only if adding u to P_i^t moves the average degree of P_i^t towards the average degree of the graph. Thus, the strict LDG multiplicative forcing term forces this assignment rule to exactly balance nodes, while the second multiplicative forcing term biases the algorithm towards edge balance. While this algorithm does not guarantee strict balance on both edges and nodes simultaneously, it does balance these well empirically.

5.4.2 Stratified balance

Optimizing against multiple constraints rapidly increases the difficulty of the partitioning problem. In contrast, we introduce a restricted problem whose goal is to balance the counts of nodes from several distinct strata. This problem arises when it is important that each individual partition resembles the demographics of a full graph. For example, in social network experiments it can be important that test groups have an equal number of men and women, or have similar levels of educational attainment. If the net-

work is assortative under these demographic traits then a good partitioning algorithm risks producing slices that are very different demographically. A particularly interesting instance of this problem is creating graph partitions that share the same degree distributions (up to integer divisibility). A node's degree in the full graph G is always available to the partitioning algorithm, and it commonly relates to important demographic node attributes such as age or geography, making balancing degree distributions a good proxy for demographic balance.

While in many situations producing miniaturized, representative partitions is a natural goal, it is quite different from the goal, or output, of many graph partitioning algorithms. For example, spectral partitioning tends to produce bisections with very different degree distributions, usually with one dense connected partition containing nodes of high degree and the other with low degree nodes that were successively 'trimmed' away from the first partition [140]. Colloquially this tendency of spectral partitioning can be described as partitioning a graph into a 'hairball' and 'whiskers' [96]. For graphs that have a pronounced core-periphery structure [24], algorithms that minimize the edge cut of a partition frequently (and rightfully so) split the graph between the core and the periphery. This results in a high quality cut between partitions with very different degree distributions, and typically, very different types of nodes.

Similarly, algorithms that aim to perform community detection are frequently tested for their ability to take graphs and produce communities whose nodes are fundamentally different. Indeed, community detection algorithms frequently demonstrate their effectiveness by revealing hidden node information utilizing only network information. In this way, community detection algorithms are frequently calibrated to create the worst possible test groups, and the partitions that least resemble the graph as a whole.

In contrast, the goal of stratified graph partitioning is to produce partitions where a node's membership in a partition reveals no information of that node's strata. We

formalize the problem as:

Problem 5.4.1 (Stratified Graph Partitioning)**.** *For a graph $G = (V, E)$, where nodes belong to L disjoint strata V_i such that $\cup_{i=1}^{L} V_i = V$, partition the graph into k disjoint partitions P_j such that $\cup_{j=1}^{k} P_j = V$, maximizing the number of uncut edges $\frac{1}{2}\sum_u |P(u) \cap N(u)|$, subject to the constraints that $|P_i \cap V_j| \leq C_{ij}$ for all i, j, and constraints C_{ij}.*

Setting each $C_{ij} = \lceil \frac{|V_j|}{k} \rceil$ requires that each partition proportionally represent the distribution of the strata in the original graph.

Despite the daunting increase in constraints, it is easy to adjust both of our simple restreaming algorithms to address this problem. For LDG this simply requires keeping additional indexing, such that a node $u \in V_\ell$ is assigned to:

$$\underset{i \in \{1,\ldots,k\}}{\operatorname{argmax}} |P_i^t \cap N(u)| \left(1 - \frac{x_{i,\ell}^t}{C_{i,\ell}}\right). \tag{5.8}$$

Likewise, one can adjust FENNEL's additive regularization, Equation 5.6, so that when assigning node $u \in V_\ell$, each x_i is also dependent on ℓ, becoming $x_{i,\ell}$.

Stratified graph partitioning has an interesting intersection with METIS in the high performance computing literature. Namely, in *multi-constraint graph partitioning* each node has an associated weight vector w_u, and the partitioning aims to balance the sum of these weights for each partition [80]. The primary aim of multi-constraint graph partitioning in the context of high performance computing is to enable efficient parallelization of large computations by dividing meshes into partitions with similar number of nodes and other attributes that affect either memory or computational requirements. The multi-constraint graph partitioning approach can apply to the stratified graph partitioning problem as well: simply consider a vertex of strata j as having weight $w_u[j] = 1$ and $w_u[i] = 0$ otherwise.

However, the added generality of multi-constraint partitioning leads the METIS implementation to have a memory footprint of $\tilde{O}(m + Ln)$, as it stores each node's weight

vector in memory. Meanwhile, the modified restreaming version of LDG requires only memory $\tilde{O}(n + Lk)$. While in the high performance computing literature, it may not be necessary to have large L, if the goal is to match degree distributions it is frequently desirable to have L on the order of $\Theta(\frac{m}{n})$, at which point L has a large impact on the runtime of METIS. For example, whereas METIS was able to partition the LiveJournal graph with 9GB of RAM, when doing 100 degree strata the memory footprint rises to 23GB. Meanwhile, the memory footprint of LDG barely changes as the number of strata are increased.

Finally, there is an important difference in emphasis between multi-constraint graph partitioning and stratified graph partitioning. While multi-constraint graph partitioning can perform stratified graph partitioning, it does so by balancing marginalized traits and not joint constraints. One must be careful of this distinction lest one may balance gender and degree by stacking one partition with high degree women and low degree men, and the other with high degree men and low degree women. This would not produce slices with comparable composition. Instead, one should make a Cartesian product of the features so that each combination of features belongs to a distinct strata.

When it is important to do streaming or restreaming multi-constraint partitioning instead of stratified partitioning, we note that the framework of Equation 5.7 in Section 4.1 can be adjusted to allow multiple constraints, though we do not examine this in our results.

5.5 Results

We now examine the performance of our restreaming partitioning algorithms on ten empirical graphs: six social graphs and four web graphs, listed in Table 1. All our graphs were obtained from the SNAP repository [139] except for the Orkut graph [110]. Graphs were made undirected by reciprocating all arcs. Self-loops and nodes with degree zero

Graph	$\|V\|$	$\|E\|$	avg deg	LDG	reLDG	reFENNEL	reFENNEL parallel	METIS(1.001)	METIS(1.03)
wikivote	7115	100762	28.32	0.867	0.775	**0.685**	0.775	0.822	0.764
astro-ph	18771	198050	21.10	0.623	0.439	**0.413**	0.438	0.535	0.372
enron	36692	183831	10.02	0.664	0.490	**0.471**	0.482	0.855	0.411
slashdot	77360	469180	12.12	0.821	0.730	**0.673**	0.686	0.711	0.693
livejournal	4846609	42851237	17.68	0.561	0.390	0.328	0.351	**0.309**	0.301
orkut	3072441	117185083	76.28	0.645	0.428	0.421	0.585	**0.376**	0.353
web-nd	325729	1090108	6.69	0.313	0.128	0.121	0.181	**0.036**	0.036
web-stanford	281903	1992636	14.13	0.378	0.207	0.176	0.237	**0.123**	0.114
web-berkstan	685230	6649470	19.41	0.341	0.203	0.188	0.283	**0.117**	0.111
web-google	875713	8644106	19.74	0.290	0.163	0.160	0.206	**0.009**	0.008

Table 5.1: The percentage of edges cut (lower is better) for the basic methods studied in this work applied to a diverse collection of graphs partitioned into 40 different partitions. The restreamed methods were run for 10 restreaming iterations while the parallel versions were split across 30 workers and run for 30 restreaming iterations. METIS(1.03) is run with 3% slack, while METIS(1.001) is run with slack 0.1% slack. For each graph the best score excluding METIS(1.03) is bolded.

were removed in order to aid the interpretability of the fraction of edges cut by a partitioning algorithm.

The densest graph we analyze here was the Orkut graph, with 3.1 million nodes and 117 million edges. The algorithms we discuss scale effortlessly beyond this size, but we are not able to analyze graphs larger than this in comparison to METIS — performing ordinary node balanced graph partitioning on Orkut in METIS already requires 18 GB of RAM, making larger graphs intractable. For the Orkut graph our restreaming LDG algorithm utilizes just 200 MB of RAM for the same graph. Since the average degree of the Orkut graph is 76, this is very nearly the expected factor of 76 times smaller. In order to compare our results to METIS we focus our analysis on graphs up to this size.

5.5.1 Node balance results

We begin by discussing our performance for the standard node balanced partitioning problem. When evaluating single-shot streaming graph partitioning algorithms, it is unclear if the gap in quality between streaming and offline algorithms should be attributed to the limitations of the single-shot view of the graph or attributed to the limited local means of the algorithm. After examining the performance of our restreaming algo-

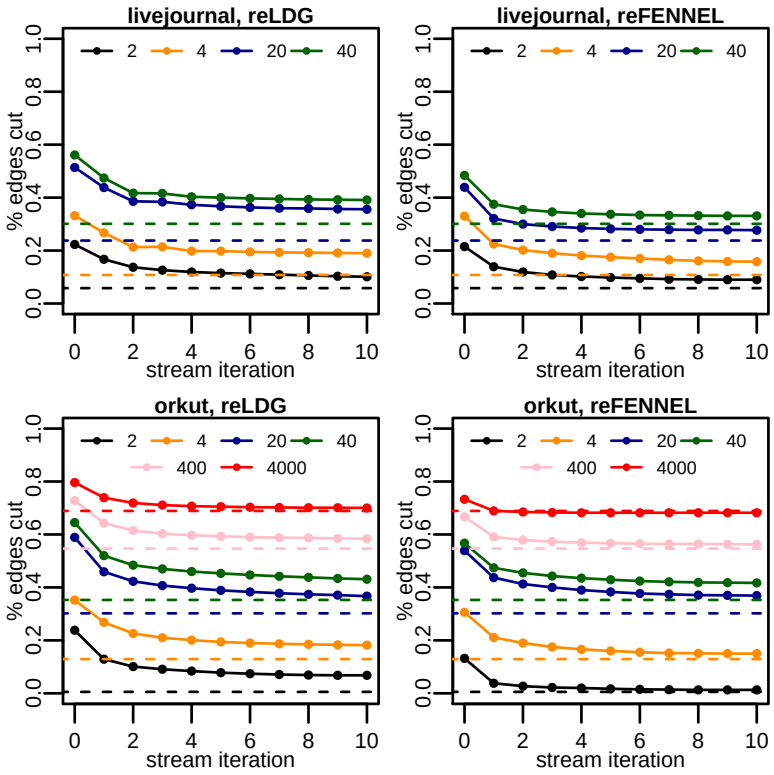

Figure 5.1: Iterating the restreaming partitioning process for static LiveJournal and Orkut graphs. The left column reports results for restreaming LDG, the right column for restreaming FENNEL. Dashed lines are METIS. Iteration zero corresponds to single-shot streaming implementations, though note that FENNEL does not guarantee balance until its final iteration due to ongoing tempering.

rithms, it is clear that much of the gap can be attributed to the limits of the single-shot view, not to a fundamental limitation of local algorithms.

Both restreaming LDG and tempered FENNEL were effective on all the graphs. However, in examining our results, it is important to distinguish between 'web' graphs, whose structure derives from the structure of hyperlinks on the internet, and 'social' graphs, whose structures represent relationships people create between each other. Indeed, there are well known differences between the structure of web and social graphs, both in their degree distribution, effective diameter, their clustering coefficients [151] and in their compressibility [37]. Social graphs are known to have significantly higher

average local clustering coefficient, indicating that the structure around individual nodes is far from divisible, while web graphs are known to compress much better than social graphs.

Consistent with these observations, web graphs have extremely high quality cuts, as seen in Table 5.1, with METIS dividing web-google into 40 partitions cutting fewer than 1% of the edges. Indeed, the multiple stages of METIS are very well suited to discovering the extremely high quality cuts of such modular graphs. Meanwhile, the highly local nature of the restreaming graphs prevents them from discovering the same high quality partitioning on web graphs that METIS is able to find. On the other hand, the dense local structure of social graphs coupled with the apparent lack of the same modular organization inherent in the web is better suited to the restreaming graphs. As such, all the restreaming algorithms are competitive with METIS on the social graphs, see Figure 5.1. In particular, restreaming FENNEL performs the best of the restreaming algorithms, out-competing METIS with 0.1% slack on four graphs and even METIS with 3% slack on two graphs. We emphasize that restreaming tempered FENNEL is finding exactly balanced partitions and using only $O(n)$ memory.

Furthermore, over the course of iterating restreams, we observe that both restreaming LDG and FENNEL converge rapidly, and at times exponentially, as seen in Figure 5.1. This provides an advantageous tradeoff between computational work and the quality of the cut. An exponential convergence rate towards the local optima is consistent with the view that during each streaming pass of the algorithm, nodes are placed permanently in their final position with independent probability p and in a transient position with probability $(1-p)$. Thus, after r restreams only $(1-p)^r$ edges remain in a transient assignment. The details of this view are not reflected in the actual microstructure of any of the resteaming results we observe, but we believe that this observation provides a helpful intuition for how restreaming algorithms attempt to correct mistakes from previous

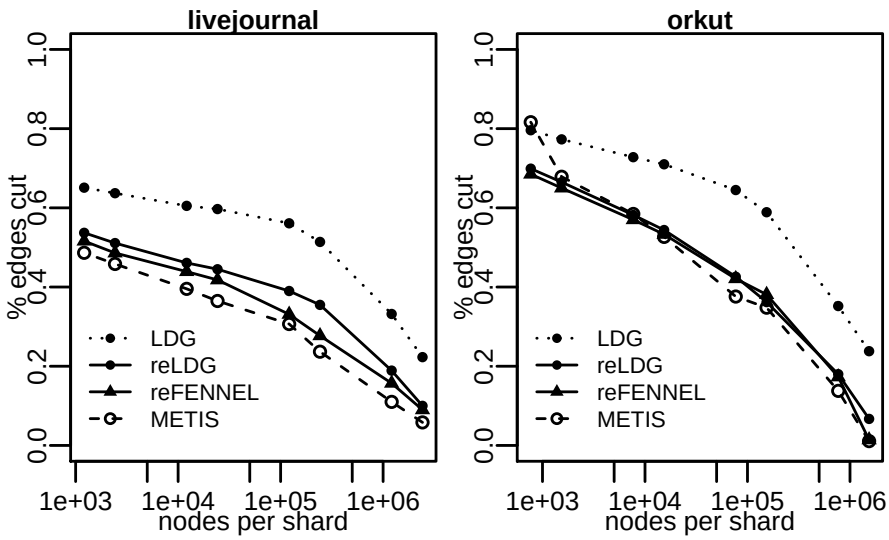

Figure 5.2: The quality of partitions as a function of the number of nodes per shard, for the LiveJournal and Orkut graphs. Notice how the restreamed algorithms essentially match METIS.

iterations.

Note that all the results for FENNEL in this section are for tempered FENNEL, and as such, node balance is only guaranteed at the end of the final iteration, so the flat performance of tempered FENNEL over the course of the many iterations in Figure 5.1 hides the fact that the algorithm is maintaining the quality of the partitioning while moving towards balance. Indeed, for some graphs, in order to achieve balance, restreaming FENNEL must decrease the quality of the partitioning during the tempering process.

As a last look at node balance, we report the results of partitioning two large graphs, LiveJournal and Orkut, into many many partitions. In Figure 5.2 we observe that our restreaming algorithms match METIS in performance across the full range of partition counts. In fact, when METIS is run with 0.1% slack and $k \geq 200$, the quality of the partitioning deteriorates rapidly, making it significantly worse than the quality of LDG and FENNEL. Since tight slack was not the intended use case for METIS, we report our results for METIS using 1% slack. Still, we see in Figure 5.2 that when Orkut is divided

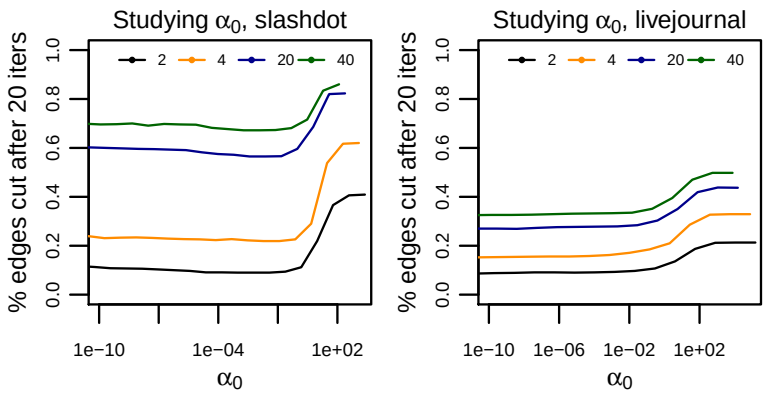

Figure 5.3: The effect of varying α_0 when tempering, where α_0 is the initial value of α and α is increased to the critical α_c over 20 restreams. For $k = 2, 4, 20, 40$ shards, we see that for sufficiently small α_0 the quality of the edge cut does not depend much on the initial α_0 from which the tempering begins.

into 4000 partitions (of roughly 1000 nodes each), the quality decreases markedly. This deterioration in quality does not occur at 4000 partitions if the slack setting for METIS is further increased.

5.5.2 Tempering

The FENNEL algorithm is only able to achieve high quality cuts with exact balance because restreaming allows for tempering. Figure 5.3 displays the effect that the initial choice of α_0 has when tempering FENNEL over 20 restreams to the critically stable α, α_c discussed in Proposition 1. Figure 5.3 shows that when tempering FENNEL, the final quality of the ultimate tempered partitioning is only sensitive to the choice of the initial α_0 when it is very large, but almost entirely insensitive to the choice as long as α_0 is small enough. Note that during a single stream, the choice of α can have a very large impact on both the quality of the partition and the departure from balance, but this importance disappears when tempering. Notice also that this observation appears to apply independently over the number of partitions being sought. This fortunately

removes some of the parameter complexity inherent in a single stream of FENNEL while also allowing FENNEL to achieve exact balance.

5.5.3 Other types of balance

In Section 4 we developed a range of different balance constraints that restreaming partitioning could be adopted towards. Here we report on the quality of the graph cuts obtained when restreaming algorithms are applied towards balance constraints other than node count. A discussion of stratified graph partitioning follows.

In Figure 5.4, we observe the differences in edge balance and degree counts when running LDG under different constraints: balancing nodes, balancing edges, balancing a sum of the two, or balancing both via the multi-balance multiplicative weights developed in Section 4.1. When either the degree counts or the node counts are left unconstrained, the algorithms clearly utilize the unconstrained flexibility. Thus, in situations where balancing degrees is important, using a method designed to balance nodes would be a poor proxy for the original problem. Indeed, even balancing based on a linear function of nodes and degree fails to balance both. Alternatively, when LDG (and FENNEL, though the results are not shown) are altered to handle multiple constraints, they are able to balance both nodes and degrees for only a small cost in partition quality. The quality of the partitions is seen in Figure 5.4. It is clear from this figure that stronger notions of balance than just node balance are within reach using simple restreaming algorithms. We now turn our attention to stratified partitioning.

5.5.4 Stratified balance results

The primary goal in stratified balance is to produce partitions representative of the original degree distribution, such that the nodes of degree d_i in the original graph are split

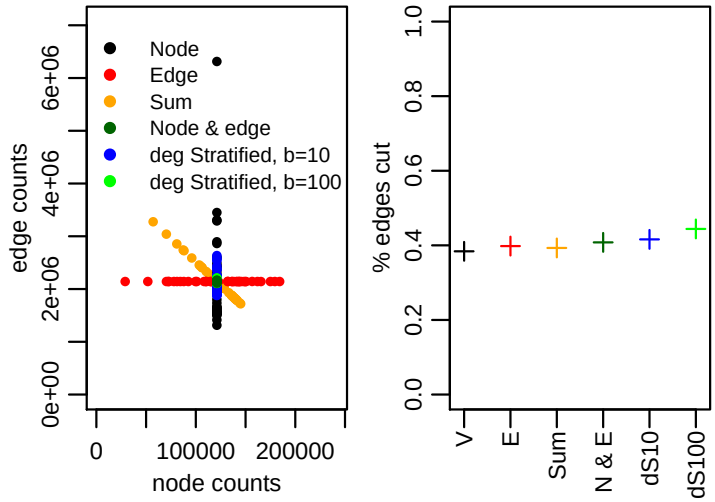

Figure 5.4: The tradeoffs between node balance and degree balance when LiveJournal is partitioned into 40 different partitions utilizing several different objectives. We see that stronger notions of balance cost very little in partition quality.

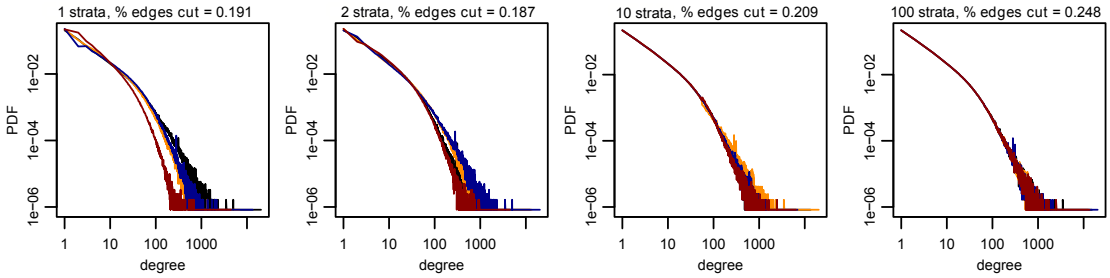

Figure 5.5: The resulting degree distribution for 4 partitions (colored differently) of LiveJournal from restreaming stratified LDG where portions of the degree distribution are explicitly balanced across 1, 2, 10 and 100 different stratified strata. As the number of strata increases the degree distributions become increasingly similar, though at a small cost in the quality of the edge cut.

equally between all the partitions. As seen in Figure 5.5, stratified LDG is able to produce partitions of increasing similarity at a small cost in quality. Since the strata in Figure 5.5 correspond to separately balancing separate contiguous degree strata, stratified graph partitioning requires that the cumulative degree distributions (CDFs) for all partitions intersect at all strata boundaries. While exactly matching the degree distribu-

tions of a graph would require as many strata as there are unique degrees, using only 100 strata produces very similar degree distributions, and even only 10 strata corrects for the majority of the difference.

Increasing the number of constraints reduces the quality of the partitioning slightly, in a manner similar to multi-constraint METIS. For a large number of strata, multi-constraint METIS and LDG produce partitionings of increasingly similar quality, such that by 100 strata on LiveJournal, METIS and LDG produce edge cuts within 1.5% of each other. Meanwhile, multi-constraints trials executed in METIS required significantly more memory than the corresponding single constraint trials, while stratified LDG only required mildly more time and memory than unstratified LDG.

5.5.5 Parallel results

Finally we consider the results of parallelizing LDG and FENNEL as discussed in Section 3.5. Figure 5.6 shows the effect of parallelizing LDG and FENNEL on the LiveJournal graph, partitioning it into 40 different partitions. Note that the first stream of the parallelized version cuts a large percentage of the graph's edges, and it takes longer for these parallelized versions to approach their final quality. However, within less than 20 restreams both algorithms, whether run on 2, 10 and 100 workers, produce partitions of quality comparable to the single thread versions of LDG and FENNEL while only requiring that $\frac{1}{2}$, $\frac{1}{10}$ and $\frac{1}{100}$ of the graph be streamed to each worker respectively. Thus, for a small price in partition quality and an increase in the number of restreams, LDG and FENNEL can be effectively parallelized to many machines. The poor partition quality after the first stream shows that this parallelization strategy can not be applied to single shot streaming partitioning, and that restreaming plays an important role in enabling parallelization.

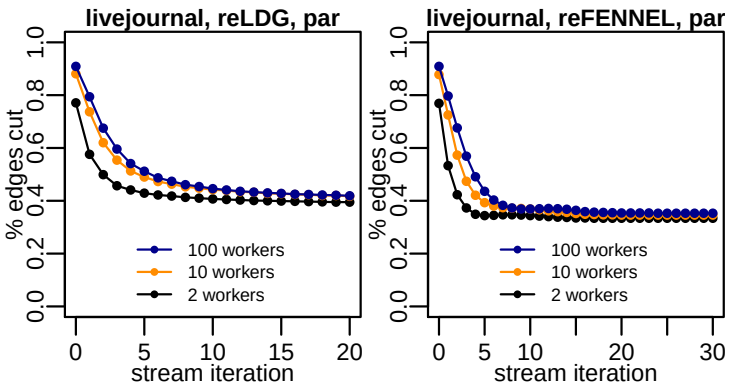

Figure 5.6: The percentage of edges cut for parallelized versions of FENNEL and LDG when partitioning LiveJournal into 40 partitions while parallelized across 2, 10 and 100 machines. Notice that while there is a small cost associated with increasing the number of *machines*, it is small compared to the gains of restreaming.

5.6 Conclusion

Given the enormous sizes of social and web graphs it is increasingly important to carefully navigate the fundamental tradeoff between the quality of a graph partition and the memory and computational requirements to compute it. To address this tradeoff, we introduce the problem of *restreaming graph partitioning* and develop two algorithms that iteratively partition graphs using only the same $O(n)$ memory required in single pass streaming graph partitioning. Surprisingly, our results demonstrate that these restreaming algorithms are able to close much of the distance between streaming graph partition algorithms and full offline graph partitioning optimization suites—at times even outperforming them. The competitiveness of these streaming graph partitions is particularly noticeable on social graphs. Furthermore, while restreaming graph partitioning preserves the same small memory footprint as single shot streaming algorithms, restreaming allows for true parallelization, with communication between workers only between streams.

The simplicity and effectiveness of these algorithms allows for their easy modification to a number of more complex objectives. In particular we introduce the problem

of *stratified graph partitioning* as a way of creating partitionings where the composition of each partition resembles the composition of the graph as a whole. Despite the significant increase in constraints in stratified graph partitioning, simple modifications to a restreaming algorithm allows for the partitioning of a large social graph such that each partition has the same degree distribution. This particular application addresses a fundamental question in the design of test groups on social graphs.

CHAPTER 6

GRAPH CLUSTER RANDOMIZATION: NETWORK EXPOSURE TO MULTIPLE UNIVERSES

A/B testing is a standard approach for evaluating the effect of online experiments; the goal is to estimate the 'average treatment effect' of a new feature or condition by exposing a sample of the overall population to it. A drawback with A/B testing is that it is poorly suited for experiments involving social interference, when the treatment of individuals spills over to neighboring individuals along an underlying social network. In this work, we propose a novel methodology using graph clustering to analyze average treatment effects under social interference. To begin, we characterize graph-theoretic conditions under which individuals can be considered to be 'network exposed' to an experiment. We then show how graph cluster randomization admits an efficient exact algorithm to compute the probabilities for each vertex being network exposed under several of these exposure conditions. Using these probabilities as inverse weights, a Horvitz-Thompson estimator can then provide an effect estimate that is unbiased, provided that the exposure model has been properly specified.

Given an estimator that is unbiased, we focus on minimizing the variance. First, we develop simple sufficient conditions for the variance of the estimator to be asymptotically small in n, the size of the graph. However, for general randomization schemes, this variance can be *lower bounded* by an *exponential* function of the degrees of a graph. In contrast, we show that if a graph satisfies a *restricted-growth condition* on the growth rate of neighborhoods, then there exists a natural clustering algorithm, based on vertex neighborhoods, for which the variance of the estimator can be *upper bounded* by a *linear* function of the degrees. Thus we show that proper cluster randomization can lead to exponentially lower estimator variance when experimentally measuring average treatment effects under interference.

6.1 Introduction

Social products and services – from fax machines and cell phones to online social networks – inherently exhibit 'network effects' with regard to their value to users. The value of these products to a user is inherently non-local, since it typically grows as members of the user's social neighborhood use the product as well. Yet randomized experiments (or 'A/B tests'), the standard machinery of testing frameworks including the Rubin causal model [133], critically assume what is known as the 'stable unit treatment value assumption' (SUTVA), that each individual's response is affected only by their own treatment and not by the treatment of any other individual. Addressing this tension between the formalism of A/B testing and the non-local effects of network interaction has emerged as a key open question in the analysis of on-line behavior and the design of network experiments [53].

Under ordinary randomized trials where the stable unit treatment value assumption is a reasonable approximation — for example when a search engine A/B tests the effect of their color scheme upon the visitation time of their users — the population is divided into two groups: those in the 'treatment' group who see the new color scheme A and those in the control group who see the default color scheme B. Assuming there are negligible interference effects between users, each individual in the treated group responds just as he or she would if the entire population were treated, and each individual in the control group responds just as he or she would if the entire population were in control. In this manner, we can imagine that we are observing results from samples of two distinct 'parallel universes' at the same time — 'Universe A' in which color scheme A is used for everyone, and 'Universe B' in which color scheme B is used for everyone — and we can make inferences about the properties of user behavior in each of these universes.

This tractable structure changes dramatically when the behavior of one user i can have a non-trivial effect on the behavior of another user j — as is the case when the

feature or product being tested has any kind of social component. Now, if i is placed in Universe A and j is placed in Universe B, then our analysis of i's behavior in A is contaminated by properties of j's behavior in B, and vice versa; we no longer have two parallel universes.

Average Treatment and Network Exposure Our goal is to develop techniques for analyzing the average effect of a treatment on a population when such interaction is present. As our basic scenario, we imagine testing a service by providing it to a subset of an underlying population; the service has a 'social' component in that i's reaction to the service depends on whether a neighbor j in the social network also has the service. We say that an individual is in the *treatment group* if the individual is provided with the service for the test, and in the *control group* otherwise. There is an underlying numerical response variable of interest (for example, the user's time-on-site in each condition), and we want to estimate the average of this response in both the universe where everyone has the service, and the universe where no one has the service, despite the fact that — since the population is divided between treatment and control — we don't have direct access to either universe.

We express this question using a formalism introduced by Aronow and Samii for causal inference without this stable unit treatment value assumption [9], with strong similarities to similar formalism introduce by Manski [105], and adapt it to the problem of interference on social networks. Let $\vec{z} \in \{0,1\}^n$ be the treatment assignment vector, where $z_i = 1$ means that user i is in the treatment group and $z_i = 0$ means the user is in the control. Let $Y_i(\vec{z}) \in \mathbb{R}$ be the potential outcome of user i under the treatment assignment vector $\vec{z}$. The fundamental quantity we are interested in is the average treatment effect, τ, between the two diametrically opposite universes $\vec{z} = \vec{1}$ and $\vec{z}' = \vec{0}$,

$$\tau(\vec{z} = \vec{1}, \vec{z}' = \vec{0}) = \frac{1}{n}\sum_{i=1}^{n}\Big[Y_i(\vec{z} = \vec{1}) - Y_i(\vec{z}' = \vec{0})\Big]. \tag{6.1}$$

This formulation contains the core problem discussed in informal terms above: unlike ordinary A/B testing, no two users can ever truly be in opposing universes at the same time.

A key notion that we introduce for evaluating (6.1) is the notion of *network exposure*. We say that i is 'network exposed' to the treatment under a particular assignment $\vec{z}'$ if i's response under $\vec{z}'$ is the same as i's response in the assignment $\vec{1}$, where everyone receives the treatment.[1] We define network exposure to the control condition analogously.

With this definition in place, we can investigate several possible conditions that constitute network exposure. For example, one basic condition would be to say that i is network exposed to the treatment if i and all of i's neighbors are treated. Another would be to fix a fraction $q > 0$ and say that i is network exposed if i and at least a q fraction of i's neighbors are treated. The definition of network exposure is fundamentally a modeling decision by the experimenter, and in this work we introduce several families of exposure conditions, each specifying the sets of assignment vectors in which a user is assumed to be 'network exposed' to the treatment and control universes, providing several characterizations of the continuum between the two universes. Choosing network exposure conditions is crucial because they specify when we can observe the potential outcome of a user as if they were in the treatment or control universe, without actually placing all users into the treatment or control universe.

Graph Cluster Randomization Following the formulation of network exposure, a second key notion that we introduce is a generic graph randomization scheme based on graph clustering, which we call *graph cluster randomization*. At a high level, graph cluster randomization is a technique in which the graph is partitioned into a set of *clus-*

[1] We also discuss adaptations to the case where the responses in these two cases differ only by a small parameter ε.

ters, and then randomization between treatment and control is performed at the cluster level. The probability that a vertex is network exposed to treatment or control will then typically involve a graph-theoretic question about the intersection of the set of clusters with the local graph structure near the vertex. We show how it is possible to precisely determine the non-uniform probabilities of entering network exposure conditions under such randomization. Using inverse probability weighting [72], we are then able to derive an unbiased estimator of the average treatment effect τ under any network exposure for which we can explicitly compute probabilities.

We motivate the power of graph cluster randomization by furnishing conditions under which graph cluster randomization will produce an estimator with asymptotically small variance. First, we observe that if the graph has bounded degree and the sizes of all the clusters remain bounded independent of the number of vertices n, then the estimator variance is $O(1/n)$, a simple but illustrative sufficient condition for smallness. The key challenge is the dependence on the degrees — in general, a collection of bounded-size clusters can produce a variance that grows exponentially in the vertex degrees. More precisely, when performing graph cluster randomization with single-vertex clusters, the variance of the estimator admits a *lower bound* that depends *exponentially* on the degrees. This raises the important algorithmic question of how to choose the clustering: bounded-size clusters provide asymptotically small variance in the number of vertices n, but if the clusters are not chosen carefully then we get an exponential dependence on the vertex degrees which could cause the variance to be very large in practice.

Cluster Randomization in Restricted-Growth Graphs We identify an important class of graphs, which we call *restricted-growth graphs*, on which a non-trivial clustering algorithm admits an *upper bound* on the estimator variance that is *linear* in the degrees of the graph. The restricted-growth condition that we introduce for graphs is an

expansion of the bounded-growth condition previously introduced for studying nearest-neighbor algorithms in metric spaces [77], designed to include low-diameter graphs in which neighborhoods can grow exponentially. Formally, let $B_r(v)$ be the set of vertices within r hops of a vertex v; our restricted-growth condition says that there exists a constant κ, independent of the degrees of the graph, such that for all vertices v and all $r > 0$, we have $|B_{r+1}(v)| \leq \kappa |B_r(v)|$. Note the comparison to the standard bounded-growth definition, which requires $|B_{2r}(v)| \leq \kappa |B_r(v)|$, a much stronger condition and not necessary for our results to hold.

For restricted-growth graphs, we provide a clustering algorithm for which the estimator variance grows only linearly in the degree. The challenge is that the variance can grow exponentially with the number of clusters that intersect a vertex's neighborhood; our approach is to form clusters from balls of fixed radius grown around a set of well-separated vertices. The restricted growth condition prevents balls from packing too closely around any one vertex, thus preventing vertex neighborhoods from meeting too many clusters. We note that for the special case of restricted-growth graphs that come with a uniform-density embedding in Euclidean space, one can use the locations of vertices in the embedding to carve up the space into clusters directly; the point, as in work on the nearest-neighbor problem [77], is to control this carving-up at a graph-theoretic level rather than a geometric one, and this is what our technique does.

Our class of restricted-growth graphs provides an attractive model for certain types of real-world graphs. Restricted-growth graphs include graphs for which there exists an embedding of the vertices with approximately uniform density in a Euclidean space of bounded dimension, such as lattices or random geometric graphs, where edges connect neighbors within some maximal metric distance.

Summary Our work thus occupies a mediating perch between recent work from the statistical literature on causal inference under interference [147, 9, 146], as well as recent

work from the computer science literature on network bucket testing [12, 81]. Our contribution extends upon the ordinary inference literature by developing exposure models and randomization schemes particularly suited for experiments on large social graphs, also showing how previous approaches are intractable. Meanwhile, we show that reducing estimator variance involves non-trivial graph-theoretic considerations, and we introduce a clustering algorithm that improves exponentially on baseline randomization schemes. Our contribution also connects to existing work on network bucket testing by contributing an exposure framework for the full graph and a randomization scheme that is capable of considering multiple exposure conditions at once, a necessity for true concurrent causal experimentation.

In Section 2 we describe our models of network exposure. In Section 3 we present our graph cluster randomization scheme, an algorithm for efficiently computing exposure probabilities, and an unbiased estimator of average treatment effects under graph cluster randomization. In Section 4 we introduce restricted-growth graphs, and show how the estimator has a variance that is linearly bounded in degree for such graphs. Section 5 concludes.

6.2 Network exposure models

For A/B randomized experiments, the *treatment condition* of an individual decides whether or not they are subject to an intervention. This typically takes two values: 'treatment' or 'control'. In most randomized experiments, the experimenter has explicit control over how to randomize the treatment conditions, and generally individuals are assigned independently. Meanwhile, the *exposure condition* of an individual determines how they experience the intervention in full conjunction with how the world experiences the intervention. Without the stable unit treatment value assumption, at worst each of the 2^n possible values of $\vec{z}$ define a distinct exposure condition for each user. Aronow and

Samii call this "arbitrary exposure" [9], and there would be no tractable way to analyze experiments under arbitrary exposure.

Consider the potential outcomes for user i. In the "arbitrary exposure" case, $Y_i(\vec{z})$ is completely different for every possible $\vec{z}$. This means that we will never be able to observe $Y_i(\vec{z})$ for either $\vec{z} = \vec{1}$ or $\vec{z} = \vec{0}$ without putting all users into the treatment or control universes. Thus, to make progress on estimating the average treatment effect under any other conditions, we require further assumptions. We do this here by assuming that multiple treatment vectors $\vec{z}$ can map to the same potential outcomes: essentially, as long as treatment vectors $\vec{z}$ and $\vec{z}'$ are "similar enough" from the perspective of a vertex i, in a sense to be made precise below, then i will have the same response under $\vec{z}$ and $\vec{z}'$.

Specifically, let σ_i^x be the set of all assignment vectors $\vec{z}$ for which i experiences outcome x. We refer to σ_i^x as an *exposure condition* for i; essentially, σ_i^x consists of a set of assignment vectors that are "indistinguishble" from i's point of view, in that their effects on i are the same. Our interest is in the particular exposure conditions σ_i^1 and σ_i^0, which we define to be the sets that contain $\vec{z} = \vec{1}$ and $\vec{z} = \vec{0}$ respectively. In this way, we are assuming that for all $\vec{z}_1 \in \sigma_i^1$, we have $Y_i(\vec{z} = \vec{z}_1) = Y_i(\vec{z} = \vec{1})$, and for all $\vec{z}_0 \in \sigma_i^0$, we have $Y_i(\vec{z} = \vec{z}_0) = Y_i(\vec{z} = \vec{0})$.[2] Note that it is possible that $\vec{z} = \vec{1}$ and $\vec{z} = \vec{0}$ belong to the same exposure condition and that $\sigma_i^1 = \sigma_i^0$, which corresponds to a treatment that has no effects.

We define an *exposure model* for user i as a set of exposure conditions that completely partition the possible assignment vectors $\vec{z}$. The set of all models, across all users, is the exposure model for an experiment. For our purposes though, it is unnecessary to entirely specify an exposure model, since we are only trying to determine the

[2] If this strikes the reader as too restrictive a definition of "exposure condition", consider instead partitioning the space of potential outcomes (rather that partitioning the space of assignment vectors) using small ϵ-sized bins, and define the "exposure conditions" as all assignment vectors that produce a potential outcome in that ϵ bin. In cases where no other potential outcomes correspond to the outcomes for $\vec{z} = \vec{0}$ or $\vec{z} = \vec{1}$, it may be more appropriate to manage bias using ϵ distances on potential outcomes this way.

average treatment effect between the extreme universes. We only care about the exposure conditions σ_i^1 and σ_i^0 for which each user i experiences exposure to the treatment or control universe[3].

Of course, the *true* exposure conditions σ_i^1 and σ_i^0 for each user are not known to the experimenter a priori, and analyzing the results of an experiment requires choosing such conditions in our framework. If the wrong exposure conditions are chosen by the experimenter, what happens to the estimate of the average treatment effect? If users are responding in ways that do not correspond to $\vec{z} = \vec{1}$ and $\vec{z} = \vec{0}$, we will be introducing bias into the average treatment effect. The magnitude of this bias depends on how close the outcomes actually observed are to the outcomes at $\vec{z} = \vec{1}$ and $\vec{z} = \vec{0}$ that we wanted to observe. It may even be favorable to allow such bias in order to lower variance in the results of the experiment.

Neighborhood Exposure We now describe some general exposure conditions that we use in what follows. In particular, we focus primarily on *local exposure conditions*, where two assignments are indistinguishable to i if they agree in the immediate graph neighborhood of i. We consider absolute and fractional conditions on the number of treated neighbors. Note we are not asserting that these possible exposure conditions are the *actual* exposure conditions with respect to the actual potential outcomes in an experiment, but rather that they provide useful abstractions for the analysis of an experiment, where again the degree of bias introduced depends on how well the exposure conditions approximate belonging to the counterfactual universes.

- *Full neighborhood exposure:* Vertex i experiences full neighborhood exposure to a treatment condition if i and all i's neighbors receive that treatment condition.

[3]If one was to assume functional relationships between the potential outcomes in different exposure conditions then other exposure conditions besides σ_i^1 and σ_i^0 could become relevant.

- *Absolute k-neighborhood exposure:* Vertex i of degree d, where $d \geq k$, experiences absolute k-neighborhood exposure to a treatment condition if i and $\geq k$ neighbors of i receive that treatment condition.
- *Fractional q-neighborhood exposure:* Vertex i of degree d experiences fractional q-neighborhood exposure to a treatment condition if i and $\geq qd$ neighbors of i receive that treatment condition.

The k-absolute and q-fractional neighborhood exposures can be considered relaxations of the full neighborhood exposure for vertex i in that they require fewer neighbors of i to have a fixed treatment condition for i to be considered as belonging to that exposure condition. In fact, the set of assignment vectors that correspond to k-absolute and q-fractional neighborhood exposures are each nested under the parameters k and q respectively. Increasing k or q decreases the set of assignment vectors until reaching full neighborhood exposure for vertex i.

It is natural to consider heterogeneous values k or q — values that differ for each user – but we limit our discussion to exposure conditions that are homogeneous across users as much as possible. We do incorporate a mild heterogeneity in the definition of k-neighborhood exposure when vertices have degree $d < k$: for these vertices we consider full neighborhood exposure instead. Fractional exposure does not require this adjustment.

Core Exposure Full neighborhood exposure is clearly only an approximation of full immersion in a universe. Beyond local exposure conditions, we also consider exposure condition with global dependence. As one approach, consider individuals as exposed to a treatment only if they are sufficiently surrounded by sufficiently many treated neighbors who are in turn also surrounded by sufficiently many treated neighbors, and so on. This recursive definition may initially appear intractable, but such recursive exposure

can in fact be characterized precisely by analyzing the k-core — and more generally the heterogeneous k-core — on the induced graph of treatment and control individuals.

Recall that the k-core of a graph $G = (V, E)$ is the maximal subgraph of G in which all vertices have degree at least k [22]. Similarly, the heterogeneous $\mathbf{k}$-core of a graph $G = (V, E)$, parameterized by a vector $\mathbf{k} = (k_1, \ldots, k_{|V|})$, is the maximal subgraph $H = (V', E')$ of G in which each vertex $v_i \in V'$ has degree at least k_i [31]. Using the definition of heterogeneous $\mathbf{k}$-core, we introduce the following natural fractional analog.

Definition 6.2.1 (Fractional q-core)**.** *The fractional q-core is the maximal subgraph $H = (V', E')$ of $G = (V, E)$ in which each vertex $v_i \in V'$ is connected to at least a fraction q of the vertices it was connected to in G. Thus, for all $v_i \in V'$, $\deg_H(v_i) \geq q \deg_G(v_i)$. Equivalently, if d_i is the degrees of vertex i, the fractional q-core is the heterogenous $\mathbf{k}$-core of G for $\mathbf{k} = (qd_1, \ldots, qd_{|V|})$.*

Since the heterogeneous $\mathbf{k}$-core is a well-defined object, so is the fractional q-core. Using this definition, we now define exposure conditions that are all stricter versions of corresponding earlier neighborhood conditions.

- *Component exposure:* Vertex i experiences component exposure to a treatment condition if i and all of the vertices in its connected component receive that treatment condition.
- *Absolute k-core exposure:* Vertex i with degree $d \geq k$ experiences absolute k-core exposure to a treatment condition if i belongs to the k-core of the graph $G[V']$, the subgraph of G induced on the vertices V' that receive that treatment condition.
- *Fractional q-core exposure:* Vertex i experiences fractional q-core exposure to a treatment condition if i belongs to the fractional q-core of the graph $G[V']$, the subgraph of G induced on the vertices V' that receive that treatment condition.

Component exposure is perhaps the strongest requirement for network exposure imaginable, and it is only feasible if the interference graph being studied is comprised of many disconnected components. We include it here specifically to note that the fractional q-core exposure for $q = 1$ reduces to component exposure. Again like the neighborhood exposure case, absolute core exposure requires heterogeneity in k across users for it to be a useful condition for all users. A parsimonious solution analogous to the solution for k-neighborhood exposure may be to consider heterogeneous max(degree, k)-core exposure. Fractional q-core exposure, like fractional q-neighborhood exposure, is again free from these parsimony problems.

Core exposure conditions are strictly stronger than the associated neighborhood exposure conditions above. In fact, every assignment vector in which a vertex i would be component or core exposed corresponds to neighborhood exposure, but not vice versa. So the assignment vectors of core and component exposure are entirely contained in those of the associated neighborhood exposure.

Other Exposure Conditions Other exposure conditions may prove relevant to particular applications. In particular, we draw attention to the intermediate concept of placing absolute or fractional conditions on the population of vertices within h hops, where $h = 1$ is the neighborhood exposure conditions above. We also note that on social networks with very high degree, for many applications it may be more relevant to define the exposure conditions in terms of a lower degree network that considers only stronger ties.

6.3 Randomization and estimation

Using the concept of network exposure, we can now consider estimating the average treatment effect τ between the two counterfactual universes using a randomized experi-

ment. Recall that $\vec{z}$ is the treatment assignment vector of an experiment. To randomize the experiment, let $\vec{z}$ be drawn from Z, a random vector that takes values on $\{0,1\}^n$, the range of $\vec{z}$. The distribution of Z over $\{0,1\}^n$ given by $\Pr(Z = \vec{z})$ is what defines our randomization scheme, and it is also exactly what determines the relevant probabilities of network exposure. For a user i, $\Pr(Z \in \sigma_i^1)$ is the probability of network exposure to treatment and $\Pr(Z \in \sigma_i^0)$ is the probability of network exposure to control.

In general, these probabilities will be different for each user and each treatment condition, and knowing these probabilities makes it possible to correct for allocation bias during randomization. In particular, it becomes possible to use the Horvitz-Thompson estimator, $\hat{\tau}$, to obtain an unbiased estimate of τ, here given by

$$\hat{\tau}(Z) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i(Z)\mathbf{1}[Z \in \sigma_i^1]}{\Pr(Z \in \sigma_i^1)} - \frac{Y_i(Z)\mathbf{1}[Z \in \sigma_i^0]}{\Pr(Z \in \sigma_i^0)}\right), \tag{6.2}$$

where $\mathbf{1}[x]$ is the indicator function. Assuming the probabilities are positive, the expectation over Z clearly gives τ, though note that this does assume that the exposure conditions are not misspecified.

Let us examine the exposure probabilities for the simplest network exposure condition, full neighborhood exposure, and under the simplest randomization scheme — independent vertex randomization, in which each vertex is independently assigned to treatment or control. If all vertices are treated independently with probability $p \in (0,1)$ then the probability of full neighborhood exposure to treatment for a user i of degree d_i is simply given by $\Pr(Z \in \sigma_i^1) = p^{d_i+1}$, and the probability of full neighborhood exposure to control is given by $\Pr(Z \in \sigma_i^0) = (1-p)^{d_i+1}$. This highlights the main challenge of network exposure: the chance that a vertex with high degree manages to reach full neighborhood exposure, or anywhere near it, can be exponentially small in d_i. Intuitively, such small exposure probabilities will dramatically increase the variance of the Horvitz-Thompson estimator, and it indicates the necessity of using more intelligent randomization.

To reduce the variance of this Horvitz-Thompson estimator, we introduce a general *graph cluster randomization* approach, creating graph clusters and randomizing assignment at the cluster level rather than at the vertex level, with clusters assigned independently. Connected vertices will then be assigned to the same treatment condition more often than would happen with independent assignment, increasing the expected number of users who are network exposed to a condition at the cost of increased correlations between users' exposure conditions.

For clarity when discussing clustering, we introduce some notation. Let the vertices be partitioned into n_c clusters $C_1, \ldots, C_{n_c}$. Let $N_i \subseteq V$ denote the neighbors of i in the graph G, and let $S_i = \{C_j : (i \cup N_i) \cap C_j \neq \emptyset\}$ denote the set of clusters that contain i or a neighbor of i; we call S_i the set of clusters to which i is *connected*. Using this notation, we will now examine the probabilities of different network exposures.

For the general creation of clusters we defer to the literature on algorithms for graph partitioning (see Chapters 4 and 5) and community detection [55]. In Section 6.4 we describe a particular algorithm for clustering graphs that satisfy a restricted-growth condition. The remainder of this section, however, describes the behavior of an arbitrary clustering on an arbitrary graph.

6.3.1 Exposure probabilities

We now examine how the probabilities of network exposure can be computed given a clustering. As a simple example, for the full neighborhood exposure condition, the probability of network exposure to treatment simply becomes $\Pr(Z \in \sigma_i^0) = p^{|S_i|}$ and to control becomes $\Pr(Z \in \sigma_i^1) = (1-p)^{|S_i|}$. We now show that computing the exposure probabilities for absolute and fractional neighborhood exposure conditions is tractable as well.

Consider the challenge of computing the probability that vertex i with degree d_i is

treated and more than k of its neighboring vertices are treated under cluster randomization. This applies when considering both absolute and fractional neighborhood exposures. First, let us reindex the clusters such that if i is connected to $|S_i| = s$ clusters, i itself resides on cluster s, and we let $j = 1, \ldots, s-1$ denote the other connected clusters. Let $w_{i1}, \ldots, w_{is}$ be the number of connections i has to each cluster, and let the Bernoulli(p) random variables $X_1, \ldots, X_s$ denote the independent coin tosses associated with each cluster. Then:

$$\begin{aligned}
\text{Pr}[Z \in \sigma_i^1] =& \text{Pr}\left[X_s = 1\right] \cdot \text{Pr}\left[\textstyle\sum_{j=1}^{s-1} w_{ij} X_j \geq k - w_{is}\right], \\
\text{Pr}[Z \in \sigma_i^0] =& \text{Pr}\left[X_s = 0\right] \cdot \text{Pr}\left[\textstyle\sum_{j=1}^{s-1} w_{ij} X_j \leq d_i - k\right].
\end{aligned}$$

Here the random quantity $\sum_j w_{ij} X_j$ obeys a weighted equivalent of a Poisson-binomial distribution, and the probabilities in question can be computed explicitly using a dynamic program defined by the following recursion

$$\begin{aligned}
\text{Pr}\left[\textstyle\sum_{j=1}^{s} w_j X_j \geq T\right] = \;& p\text{Pr}\left[\textstyle\sum_{j=1}^{s-1} w_{ij} X_j \geq T - w_{is}\right] + \\
& (1-p)\text{Pr}\left[\textstyle\sum_{j=1}^{s-1} w_{ij} X_j \geq T\right].
\end{aligned}$$

Note that T is bounded by the maximum vertex degree d_{max}, making this a polynomial time dynamic program with runtime $O(d_{max} s)$. We formalize this computation into the following proposition.

Proposition 6.3.1. *The probability that vertex i is treated and $\geq k$ neighboring vertices are treated under independent cluster randomization is given by*

$$\textit{Pr}[Z \in \sigma_i^1] = pf(s-1, k-w_{is}; p, \vec{w})$$

where

$$\begin{aligned}
f(1, T; p, \vec{w}_i) &= p\mathbf{1}[T < w_{i1}], \\
f(j, T; p, \vec{w}_i) &= pf(j-1, T - w_{ij}; p, \vec{w}_i) \\
&\quad + (1-p) f(j-1, T; p, \vec{w}_i).
\end{aligned}$$

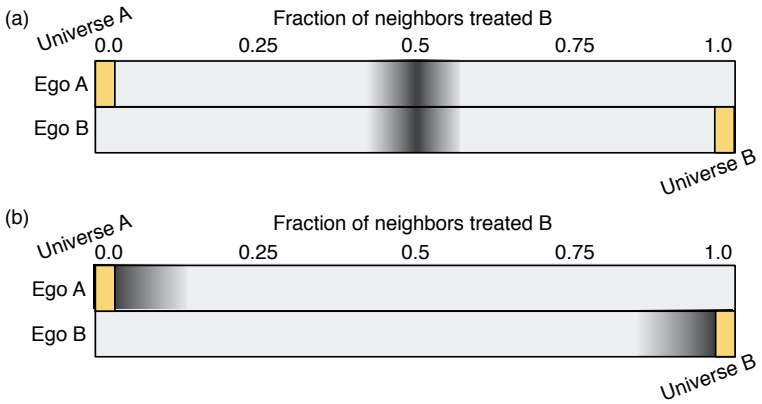

Figure 6.1: The probability distribution over the exposure space for a single individual, where the exposure conditions σ_i^0 and σ_i^1 are shown in yellow for both (a) an i.i.d. vertex randomization and (b) an ideal cluster randomization, where the probability mass is collected at exposure conditions of interest.

The probability that vertex i is in control and $\geq k$ neighboring vertices are in control under independent cluster randomization is given by

$$\boldsymbol{Pr}[Z \in \sigma_i^0] = (1-p)[1 - f(s-1, d_i - k + 1; p, \vec{w})].$$

Recall that these partial neighborhood exposure conditions (absolute and fractional) are nested. In fact, for a given vertex i the recursion can be used to derive the probability for every possible threshold value under consideration in a single $O(d_{max}s)$ double for-loop. Such a computation in fact returns the probability distribution over the exposure space for each individual. See Figure 6.1 for illustrations of what this distribution can look like.

The dynamic program above only provides a means of exactly computing exposure probabilities for absolute and fractional neighborhood exposure conditions. Unfortunately, how to efficiently compute the exact probability of k-core and fractional q-core exposure conditions is unclear, but recall that these exposure conditions were formally nested subsets of the corresponding neighborhood exposure conditions. This at least allows us to upper bound the core exposure probabilities, and we formalize this connection via the following proposition. Because we are generally concerned about exposure

probabilities being too small, this upper bound can be useful in identifying vertices with problematically small probabilities already under neighborhood exposure.

Proposition 6.3.2. *The probability vertex i is network exposed to a treatment condition under core exposure is less than or equal to the probability under the analogous neighborhood exposure:*

$$\begin{aligned} Pr(Z \in \sigma_i^x | k\text{-}core) &\leq Pr(Z \in \sigma_i^x | k\text{-}nhood), \\ Pr(Z \in \sigma_i^x | \, frac\ q\text{-}core) &\leq Pr(Z \in \sigma_i^x | \, frac\ q\text{-}nhood). \end{aligned}$$

It is possible that a useful direct estimate of the core exposure probabilities can be obtained via Monte Carlo sampling of the randomization, but we do not explore that possibility here.

6.3.2 Estimator variance

The variance of the Horvitz-Thompson estimator under interference has been studied by Aronow and Samii [9], where they also present several variance reduction schemes. Estimating the variance under their approach requires knowledge of joint exposure conditions, the joint probability that vertex i is network exposed to treatment/control and vertex j is network exposed to treatment/control. This is the probability that the random vector Z is in the exposure condition for vertex i and for vertex j simultaneously, i.e. Pr$(Z \in (\sigma_i^1 \cap \sigma_j^1))$ for joint network exposure to treatment. If one is interested in computing the variance of the estimator analytically then there is nothing fundamentally different about this probability computation when compared to the single vertex exposure probability, aside from the fact that the intersection of the two sets can be empty. Aronow and Samii observe that an empty intersection makes it impossible to derive an unbiased estimate of the variance (though they show how the variance can still be upper bounded), but it does not bias the effect estimator itself.

The variance of the effect estimator where

$$\hat{Y}^x(Z) = \frac{1}{n}\sum_i [Y_i(Z)\mathbf{1}[Z \in \sigma_i^x]/\text{Pr}(Z \in \sigma_i^x)]$$

is given by

$$\begin{aligned}\text{Var}[\hat{\tau}(Z)] &= \Big[\text{Var}[\hat{Y}^1(Z)] + \text{Var}[\hat{Y}^0(Z)] - \\ &\qquad 2\text{Cov}[\hat{Y}^1(Z), \hat{Y}^0(Z)]\Big]. \end{aligned} \tag{6.3}$$

Assuming the exposure conditions are properly specified, namely assuming that $Y_i(\vec{z})$ is constant for all $\vec{z} \in \sigma_i^x$, we can introduce the notation $Y_i(\sigma_i^x) := Y_i(\vec{z} \in \sigma_i^x)$. Using the further notation $\pi_i^x := Pr[Z \in \sigma_i^x]$ and $\pi_{ij}^{xy} := Pr[Z \in (_i^x \cup \sigma_j^y)]$ we obtain

$$\begin{aligned}\text{Var}[\hat{Y}^x(Z)] &= \frac{1}{n^2}\Bigg[\sum_{i=1}^n \frac{1-\pi_i^x}{\pi_i^x} Y_i(\sigma_i^x)^2 + \\ &\qquad \sum_{i=1}^n \sum_{\substack{j=1 \\ j\neq i}}^n \frac{\pi_{ij}^{xx} - \pi_i^x \pi_j^x}{\pi_i^x \pi_j^x} Y_i(\sigma_i^x) Y_j(\sigma_j^x)\Bigg], \end{aligned} \tag{6.4}$$

and

$$\begin{aligned}\text{Cov}[\hat{Y}^1(Z), \hat{Y}^0(Z)] = \frac{1}{n^2}\Bigg[\sum_{i=1}^n \sum_{\substack{j=1 \\ j\neq i}}^n \frac{\pi_{ij}^{10} - \pi_i^1 \pi_j^0}{\pi_i^1 \pi_j^0} Y_i(\sigma_i^1) Y_j(\sigma_j^0) - \\ \sum_{i=1}^n Y_i(\sigma_i^1) Y_i(\sigma_i^0)\Bigg]. \end{aligned} \tag{6.5}$$

The above expressions make it evident that the variance is very tightly controlled by the probabilities of exposure, and in order to upper bound the variance we will require lower bounds on the probabilities π_i^x and also upper bounds on the joint probabilities π_{ij}^{xy}, for all vertex pairs and all combinations of x and y. For neighborhood exposure, we can now write basic sufficient conditions under which the variance of the estimator is asymptotically $O(1/n)$ in n for graph cluster randomization.

Proposition 6.3.3. *Assume the potential outcomes $Y_i(\cdot)$ are all $O(1)$ in n. If G has maximum degree $O(1)$ and the size of each cluster is $O(1)$, then the variance of the*

Horvitz-Thompson estimator for full, k-neighborhood, and q-fractional neighborhood exposure under graph cluster randomization is $O(1/n)$.

Proof. Assume G has maximum degree $O(1)$ and the size of each cluster is $O(1)$. All of the single sums are clearly $O(n)$: π_i^x is $\mathrm{O}(1)$ since all vertices have bounded degree. For the double sums, note that $\pi_{ij}^{xx} = \pi_i^x \pi_j^x$ if and only if i and j have no common cluster neighbors, $|S_i \cap S_j| = 0$. Whenever $|S_i \cap S_j| > 0$, $\pi_{ij}^{xx} > \pi_i^x \pi_j^x$ for full, k-neighborhood, and q-fractional neighborhood exposure. Further, $\pi_{ij}^{10} < \pi_i^1 \pi_j^0$ if $|S_i \cap S_j| > 0$ and $\pi_{ij}^{10} = \pi_i^1 \pi_j^0$ otherwise.

So the terms of the double sums are zero whenever $\pi_{ij} = \pi_i \pi_j$ and when the terms are not zero ($|S_i \cap S_j| > 0$), they are all positive and bounded above $O(1)$ due to the bounded degrees. We now bound the number of vertices j for which $|S_i \cap S_j| > 0$. Vertex i at most connects to $O(1)$ clusters and therefore $|S_i| = O(1)$. For all $C \in S_i$, we have that $|S_i \cap S_j| > 0$ for any $j \in C$ and for any vertex j that is adjacent to a vertex in cluster C. Both of these contributions is $O(1)$, giving an $O(1)$ contribution of vertices for each $C \in S_i$. Since there are $O(1)$ such clusters, this is still $O(1)$ vertices j in total for vertex i such that $|S_i \cap S_j| > 0$. Thus for each vertex, at most $O(1)$ of the terms in the double sum are positive, making the total variance $O(1/n)$. □

The strength of this general result is that it achieves an $O(1/n)$ bound on the variance when the maximum degree is bounded. The problem is that the variance can grow exponentially in the degrees of the graph. In this next section we address this issue, introducing a condition on a graph that ensures we can find a clustering into sets of size $O(1)$ — consistent with the above result – for which the variance grows as $O(1/n)$ but is also linear rather than exponential in the maximum degree.

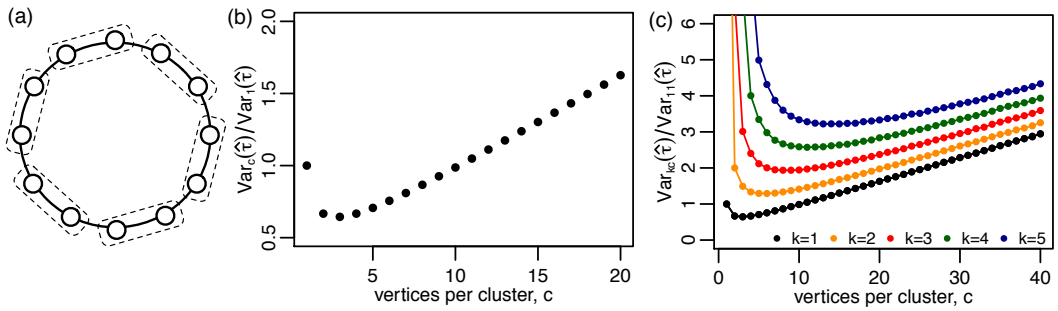

Figure 6.2: The cycle graph, (a) where vertices respond $\bar{Y}$ to treatment and 0 to control, shown clustered in groups of $c = 2$ vertices. (b) Asymptotic variance of the estimator for this graph as a function of the number of vertices per cluster, normalized by estimator variance for $c = 1$ vertices per cluster. (c) Simulated variance of the estimator for kth powers of the cycle graph for $k = 1, \ldots, 5$ as a function of the number of vertices per cluster. For each k the variance for cluster size $c = 2k + 1$ grows linearly in k.

6.4 Variance on restricted-growth graphs

In order to measure average treatment effects under interference on large-scale graphs, it is necessary to design a randomization scheme capable of containing the estimator variance for high-degree vertices. In this section we show that any graph satisfying our restricted-growth condition admits a clustering that can produce an unbiased effect estimate that is both $O(1/n)$ and linear in the degrees of the graph. In contrast, we show that with less careful clustering, it is easy for the variance to grow exponentially in the degrees.

Let us first define restricted-growth graphs. Let $B_r(v)$ be the set of vertices within r hops of a vertex v.

Definition 6.4.1. *A graph $G = (V, E)$ is a restricted-growth graph if for all vertices $v \in V$ and all $r > 0$, we have $|B_{r+1}(v)| \leq \kappa |B_r(v)|$.*

As mentioned in the introduction, graphs derived from a uniform-density embedding in a Euclidean space of dimension m exhibit restricted growth, with growth constant $\kappa = 2^m$ independent of degree. To develop intuition for the restricted-growth assumption, we

first analyze the variance using graph cluster randomization on a family of particularly tractable restricted-growth graphs, kth powers of the cycle. We follow this analysis by proving bounds on the variance for general restricted-growth graphs.

6.4.1 Cycle and powers of the cycle examples

First we will consider a simple graph consisting of a single cycle with n vertices. For this graph, we consider the full neighborhood exposure model, where we are interested in the average treatment effect between σ_i^1, when a vertex is treated and both of their neighbors are treated, and σ_i^0, when a vertex is not treated and neither of their neighbors are treated. For the fixed responses of the vertices to treatment and control, we assume that all vertices uniformly respond $Y_i(\sigma_i^1) = \bar{Y}$ to network exposure to the treatment and $Y_i(\sigma_i^0) = 0$ to network exposure to the control. The cycle graph clearly admits an intuitively obvious clustering using the cycle structure, with contiguous blocks of c vertices randomized together. As a last assumption, assume that clusters are selected under a balanced randomization with $p = 1/2$. Our goal is to determine how the variance of the Horvitz-Thompson average treatment effect estimator depends on the size c of these clusters. For this basic combination of graph, exposure condition, responses, and clustering, one can derive the asymptotic variance exactly.

Consider the variance presented in (6.3) above. Since all vertices respond zero to the control condition in our example, as long as the exposure probability for the control condition is strictly positive then both $\text{Var}(\hat{Y}(\sigma_0))$ and $\text{Cov}(\hat{Y}(\sigma_1), \hat{Y}(\sigma_0))$ are zero. Since our calculations will rely only on probabilities π_i^1 for the exposure to treatment condition, we omit the superscript. The variance is then:

$$\text{Var}[\hat{\tau}(Z)] = \frac{\bar{Y}^2}{n^2}\left[\sum_{i=1}^{n}\left(\frac{1}{\pi_i} - 1\right) + \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n}\left(\frac{\pi_{ij}}{\pi_i\pi_j} - 1\right)\right]. \tag{6.6}$$

Notice that the terms of the double sum are only non-zero for vertex pairs where $\pi_{ij} \neq$

$\pi_i\pi_j$.

First, consider the case of each vertex being its own cluster. The probability of being exposed and both of one's neighbors being exposed is equal to the probability of seeing three independent coins come up heads. When the randomization is balanced (e.g. $p = 1/2$), we obtain $\pi_i = 1/8, \forall i$. Note that the co-assignment probabilities depend on whether vertices i and j are neighbors or share a neighbor. From this we derive $\pi_{ij} = 1/16$ if $|i-j| = 1$ and $\pi_{ij} = 1/32$ if $|i-j| = 2$, and if $|i-j| > 2$, the probabilities are independent. We obtain $\text{Var}(\hat{\tau}(Z)) = (15/2)\bar{Y}^2\frac{1}{n} + O(1/n^2)$.

Now, consider randomizing blocks of $c \geq 2$ vertices, where c does not depend on n. The calculations for this case are expansive but straight-forward. We consider a single one of the equivalent cyclically shifted possibilities. The calculation requires handling $c = 2$ and $c \geq 3$ separately, but the expression for $c \geq 3$ as a function of c holds for $c = 2$ as well, so we omit the special case for brevity. The variance calculation depends on distance $\Delta = |i-j|$ up to $\Delta = c+1$, and for $c \geq 3$ this evaluates to:

$$\text{Var}[\hat{\tau}(Z)] = \frac{\bar{Y}^2}{n^2}\Bigg[\left(n + \frac{4n}{c}\right) + \underbrace{\frac{2n}{c}(c+2)}_{\Delta=1} + \underbrace{\frac{2n}{c}\sum_{k=2}^{c-2}(c-k+2)}_{1<\Delta<c-1} + \underbrace{\frac{2n}{c}3}_{\Delta=c-1} + \underbrace{\frac{2n}{c}2}_{\Delta=c} + \underbrace{\frac{2n}{c}}_{\Delta=c+1}\Bigg] + O\left(\frac{1}{n^2}\right).$$

This reduces to $\text{Var}(\hat{\tau}(Z)) = \left(\frac{c}{2} + 2 + \frac{4}{c}\right)\bar{Y}^2\frac{1}{n} + O(1/n^2)$, which holds for all $c \geq 2$.

Combining these calculations, the asymptotic variance of the estimator for all c is plotted in Figure 2. Notice that the variance is minimized when randomizing clusters of size $c = 3$, which corresponds exactly to the size of neighborhoods on the simple cycle.

To build upon this observation, we now examine the simulated variance for higher degree extensions of the cycle, the so-called kth power of the cycle, where analytic derivation is already unwieldy. Thus, we use a simulation of the cluster randomization procedure to examine how the variance of the effect size estimator depends on the cluster

size for these higher degree graphs.

The kth power of a cycle graph consists of a cycle where each vertex is connected to the k nearest neighbors on each side, yielding a regular graph where all vertices have degree $d = 2k$. By sampling one million cluster randomizations on graphs with $n = 5000$ vertices, we can compute the sample variance of the estimator across these samples. The results are shown in Figure 2, for $k = 1$ through $k = 5$. The simulations for $k = 1$ agree precisely with the overlaid asymptotic calculations.

Notice how the optimal cluster size c appears to scale approximately linearly in degree, and also notice how the variance at the optimal clustering size, the minimum value of each curve as k increases, appears to scale linearly in k. While the exact variance as a function of cluster size c is unwieldy to derive, we are able to provide the following upper bound, showing how the variance of the estimator for clusters of size $c = d + 1$ scales linearly in the degree d of the graph. This suggests that one should treat contiguous blocks of the cycle attuned to the size of the neighborhood of the vertices.

When deriving this upper bound, it is no longer necessary to assume a uniform response $Y_i(\sigma_i^1) = \bar{Y}$, and instead we simply assume that the responses are upper bounded by some value $Y_i(\sigma_i^1) \leq Y_M$.

When clusters have size $c = d+1$, each vertex can be connected to at most 2 clusters, meaning that $1/\pi_i \leq 1/p^2$ for all i. So

$$\text{Var}[\hat{\tau}(Z)] \quad \leq \quad \frac{Y_M^2}{n^2}\left[\sum_{i=1}^{n}(p^{-2}-1) + \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n}(\frac{\pi_{ij}}{\pi_i\pi_j}-1)\right].$$

Now each vertex has a non-independent joint assignment probability (such that $\pi_{ij} \neq \pi_i\pi_j$) with at most $3d + 1$ other vertices: up to $2d + 1$ other vertices when they are adjacent to two clusters, the $d/2$ to the left of the left cluster, and the $d/2$ to the right of the right cluster. The joint assignment probability π_{ij} is at most p^2, since two vertices can not both be at the center of a cluster. For each i, the sum indexed by j then can be

bounded, producing

$$\text{Var}[\hat{\tau}(Z)] \leq Y_M^2(p^{-2}-1)(3d+2)\frac{1}{n}.$$

This result tells us that it is possible to experimentally measure network effects on a cycle graph of very high degree d with a variance that is only linear in d, provided that the vertices are clustered in contiguous blocks of $d+1$ vertices. We now show how this strategy of bounding the variance applies to a much more general class of graphs, using a clustering algorithm that does not require knowledge of any geometric structure.

6.4.2 Clustering restricted-growth graphs

We now begin developing the main result of this section, a cluster randomization scheme for the class of restricted-growth graphs. The first component is a clustering algorithm for such graphs in which each vertex is connected to at most a constant number of clusters, independent of the degree of the vertex. This will then imply that the variance on any restricted-growth graph can be upper bounded by a function linear in the degree. Our clustering shows that the nice decomposition of the cycle by contiguous regions can be generalized to arbitrary graphs in our class. In other words, the geometry isn't crucial; the restricted-growth property is enough.

Consider a restricted-growth graph $G = (V, E)$; we will present the case in which G is d-regular, but as we note below, the regularity can be relaxed to arbitrary degree distributions at the cost of a weaker but still constant bound on the number of connected clusters.

Recall that the restricted-growth condition says there exists κ so that for all v and all $r > 0$, we have $|B_{r+1}(v)| \leq \kappa |B_r(v)|$. Importantly, $r = 0$ is different: $B_0(v)$ is the singleton set $\{v\}$, while $B_1(v)$ is the neighborhood of v and hence has size $d+1$. Thus $|B_1(v)|/|B_0(v)| = d+1$, potentially much larger than the bound of κ on the ratio

$|B_{r+1}(v)|/|B_r(v)|$ for $r > 0$. This is the crux of the restricted-growth condition: from radius 0 to 1 we have unrestricted growth (a factor of $d + 1$), but then the growth slows to factors of κ which can be bounded separately from d.

In the language of metric spaces, we will cluster the graph using a *3-net* for the shortest-path metric of G [66]. Formally, in a metric space X, an r-net $Y \subseteq X$ is a collection of points that are mutually at distance at least r from each other, but the union of all their r-balls covers the space, $X \subseteq \cup_{y \in Y} B_r(y)$. Accordingly, we call our construction a *3-net clustering* of the graph. To build a 3-net clustering, we will iteratively identify vertices $v_1, v_2, ...$, 'marking' vertices as we do this. Afterwards we will identify clusters $C_1, C_2, ...$ to go with these vertices. More explicitly, we perform the following procedure consisting of two principle stages:

- Initially all vertices are unmarked.
- While there are unmarked vertices, in step j find an arbitrary unmarked vertex v, selecting v to be vertex v_j and marking all vertices in $B_2(v_j)$.
- Suppose k such vertices are defined, and let $S = \{v_1, v_2, ..., v_k\}$.
- For every vertex w of G, assign w to the closest vertex $v_i \in S$, breaking ties consistently (e.g. in order of lowest index).
- For every v_j, let C_j be the set of all vertices assigned to v_j.

The sets $C_1, \ldots, C_k$ are then our 3-net clustering. The key property of this clustering is the following result, which establishes that each vertex is connected to a number of clusters that can be bounded by a function of κ, independent of the degree.

Proposition 6.4.2. *Consider any 3-net clustering of a graph $G = (V, E)$. For all $w \in V$, the neighborhood $B_1(w)$ has a non-empty intersection with at most κ^3 distinct clusters.*

Proof. We first claim that for all $v_j \in S$, we have $C_j \subseteq B_2(v_j)$. Indeed, consider any vertex $w \neq v_j$ in C_j. We have $w \notin S$, since otherwise w would belong to the cluster

identified with itself. Now, consider the iteration i in which w was marked; we have $w \in B_2(v_i)$. Since $w \in C_j$ and it is assigned to the closest vertex in S, it follows that $w \in B_2(v_j)$. Thus $C_j \subseteq B_2(v_j)$.

Next, we claim that for all $v_i, v_j \in S$, the sets $B_1(v_i)$ and $B_1(v_j)$ are disjoint. Suppose by way of contradiction that $B_1(v_i) \cap B_1(v_j) \neq \emptyset$. It would follow that $v_i \in B_2(v_j)$ and vice versa. But then if we consider the vertex among v_i and v_j that was added to S first, the other of v_i or v_j would have been marked in that iteration, and hence it could not have been added to S as well. This contradiction establishes that $B_1(v_i)$ and $B_1(v_j)$ are disjoint.

To complete the proof, suppose by way of contradiction that $B_1(w)$ has a non-empty intersection with more than κ^3 distinct clusters: for some $t > \kappa^3$, let $u_1, u_2, \ldots, u_t$ be distinct vertices in $B_1(w)$ and $v_{i_1}, \ldots, v_{i_t}$ be distinct vertices in S such that $u_h \in C_{i_h}$ for $h = 1, 2, \ldots, t$.

Since $C_{i_h} \subseteq B_2(v_{i_h})$, and C_{i_h} contains a vertex adjacent to w (or contains w itself), we have $v_{i_h} \in B_3(w)$, and hence $B_1(v_{i_h}) \subseteq B_4(w)$. The neighborhoods $B_1(v_{i_1}), B_1(v_{i_2}), \ldots, B_1(v_{i_t})$ are all pairwise disjoint as argued above, and they are all contained in $B_4(w)$, which implies that $|B_4(w)| \geq t(d+1) > \kappa^3(d+1)$. But applying the bounded growth inequality $|B_{r+1}(w)| \leq \kappa |B_r(w)|$ three times we have $|B_4(w)| \leq \kappa^3(d+1)$, a contradiction. This establishes that $B_1(w)$ can have a non-empty intersection with at most κ^3 distinct clusters. □

The above result is formulated for d-regular graphs. But in fact one can show a weaker bound depending only on κ as in Proposition 6.4.2 even for arbitrary restricted-growth graphs, without any requirement on the degrees. This weaker bound of κ^6 can be established by observing that any restricted-growth graph exhibits a "bounded gradient" on the vertex degrees, whereby vertices that are near each other in the graph must have similar degrees. Combining this fact with proof of Proposition 6.4.2 leads to the desired

bound.

6.4.3 Variance bounds

We now apply the above results to bound the variance of the effect estimator $\hat{\tau}$. Throughout this section we assume that all responses obey upper bounds and positive lower bounds, $Y_i^x \in [Y_m, Y_M]$ for both exposure to treatment and control, $x = 0, 1$. The reason for the positive lower bounds is that without them the users could all be responding zero to all treatments, making the variance zero regardless of the treatment scheme. We also assume the randomization probability p is not degenerate, i.e. $p \in (0, 1)$. We present the results for d-regular graphs to keep expressions manageable, but analogous results can be derived for arbitrary degrees.

We first establish an exponential lower bound for the variance under vertex-level randomization, and then we show a contrasting linear upper bound for the variance under our 3-net cluster randomization scheme.

Proposition 6.4.3. *The variance of the HT estimator under full neighborhood exposure for vertex randomization of a graph with n vertices is lower bounded by an exponential function in the degree d of the graph, Var$[\hat{\tau}(Z)] \geq O(1/n)(p^{-(d+1)} + (1-p)^{-(d+1)} - 1)$.*

PROOF. The joint assignment probabilities for two vertices having the same exposure is at least the product of their individual probabilities, $\pi_{ij}^{xx} \geq \pi_i^x \pi_j^x$ for $x = 0, 1$. Thus the double sum in equation (4) is non-negative. Similarly, for opposing exposure conditions, we have $\pi_{ij}^{xy} \leq \pi_i^x \pi_j^y$ for $x \neq y$, which makes equation (5) a non-negative contribution to equation (3). We focus our lower bound on the main term of equation (4). Inputting the probabilities $\pi_i^1 = p^{d+1}$ and $\pi_i^0 = (1-p)^{d+1}$ and lower bounding

responses gives us the desired result.

$$\begin{aligned}\text{Var}[\hat{\tau}(Z)] &\geq \frac{1}{n^2}\left[\sum_{i=1}^{n}(\frac{1}{\pi_i^1}-1)(Y_i^1)^2+\sum_{i=1}^{n}(\frac{1}{\pi_i^0}-1)(Y_i^0)^2\right] \\ &\geq \frac{Y_m^2}{n}(p^{-(d+1)}+(1-p)^{-(d+1)}-2). \qquad \square\end{aligned}$$

For graphs with arbitrary degree distributions, this bound becomes $\text{Var}[\hat{\tau}(Z)] \geq O(1/n)\sum_{i=1}^{n}(p^{-(d_i+1)}+(1-p)^{-(d_i+1)}-2)$, which is exponential in the degree of each vertex, meaning that even a single high degree vertices can easily explode the variance.

We now turn to our linear upper bound for growth-restricted graphs when using our 3-net clustering.

Proposition 6.4.4. *The variance of the HT estimator under full, q-fractional, or k-absolute neighborhood exposure for a 3-net cluster randomization of a restricted-growth graph is upper bounded by a function linear in the degree d of the graph.*

Proof. Recall that the variance of the estimator is given by: $\text{Var}(\hat{\tau}(Z)) = \text{Var}(\hat{Y}^1) + \text{Var}(\hat{Y}^0) - 2\text{Cov}(\hat{Y}^1, \hat{Y}^0)$. We begin by upper bounding the variance of $\hat{Y^1}(Z)$, and the upper bound for $\hat{Y^0}(Z)$ follows the same principle. We conclude by bounding the covariance term. By Proposition 6.4.2, each vertex is connected to at most κ^3 clusters. Thus we have the lower bound $\pi_i^1 \geq p^{\kappa^3}$, for both full and fractional neighborhood exposure.

$$\text{Var}[\hat{Y^1}(Z)] \leq \frac{Y_M^2}{n^2}\left[n(\frac{1}{p^{\kappa^3}}-1)+\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}(\frac{\pi_{ij}^1}{\pi_i^1\pi_j^1}-1)\right].$$

For each vertex i, the inner of the two sums is only nonzero at those vertices j for which the assignments are dependent. If the assignments for i and j are dependent, then they must each have neighbors in the same cluster C_h associated with a vertex v_h in the set of cluster centers. Since the proof of Proposition 6.4.2 established that $C_h \subseteq B_2(v_h)$, it follows that i and j are each within distance 3 of v_h and hence within distance 6 of each

other. Thus, any j whose assignment is dependent on i's must lie within $B_6(i)$, and so by the restricted-growth condition, there can be at most $|B_6(i)| \leq \kappa^5 |B_1(i)| = \kappa^5(d+1)$ such vertices j. Thus the sum over such j has at most $\kappa^5(d+1)$ terms. Also, $\pi_{ij}^1 \leq p$ applies, since the two vertices must depend on at least one cluster. We obtain

$$\text{Var}[\hat{Y}^1(Z)] \leq Y_M^2[(p^{-\kappa^3} - 1) + \kappa^5(d+1)(p^{-2\kappa^3-1} - 1)]\frac{1}{n}.$$

Now, consider the contribution of the covariance term to the variance, $-2\text{Cov}(\hat{Y}^1, \hat{Y}^0)$, a positive quantity. Starting from equation (6.5), we apply the upper bound for the responses Y_i to obtain

$$-2\text{Cov}[\hat{Y}^1(Z), \hat{Y}^0(Z)] \leq -\frac{2Y_M^2}{n^2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \left(\frac{\pi_{ij}^{10}}{\pi_i^1 \pi_j^0} - 1 \right) + \frac{2Y_M^2}{n}.$$

As with the previous analogous expression, for each i the inner sum is non-zero for at most $\kappa^5(d+1)$ other vertices j. For the remaining terms, the quantity $-(\pi_{ij}^{10}/(\pi_i^1 \pi_j^0) - 1)$ is trivially upper bounded by 1. Thus we obtain

$$-2\text{Cov}[\hat{Y}^1(Z), \hat{Y}^0(Z)] \leq \frac{2Y_M^2}{n}[\kappa^5(d+1) + 1].$$

Combining the upper bounds, we obtain a total upper bound that is linear in degree, as desired. □

The restricted-growth condition we used was derived for regular graphs, but as we noted earlier, for restricted-growth graphs with arbitrary degree distributions we can apply a weaker but still constant bound on the cluster dependencies to obtain a variance bound that is still linear in the degree.

6.5 Conclusion

The design of online experiments is a topic with many open directions (see e.g. [87]); in this work we have focused on the open question of A/B testing when treatment effects can spill over along the links of an underlying social network. We introduced a

basic framework for reasoning about this issue, as well as an algorithmic approach — graph cluster randomization — for designing A/B randomizations of a population when network spillover effects are anticipated. Appropriate clustering can lead to reductions in variance that are exponential in the vertex degrees. We emphasize that beyond the class of graphs where we prove bounds, graph cluster randomization is a technique that can be applied to arbitrary graphs using arbitrary community detection or graph partitioning algorithms, though we do not provide any variance bound guarantees for these scenarios.

There are many further directions for research suggested by the framework developed here. A first direction is to formulate a computationally tractable objective function for minimizing the variance of the Horvitz-Thompson estimator. One approach would be via minimizing an adversarial variance, as in [81]. Another problem that may be relevant is to find a clustering that minimizes A/A variance for full neighborhood exposure under the assumption of known control potential outcomes. Can good clusterings for A/A variance lead to good solutions for A/B testing? We note that A/A variance minimization would not be useful when the treatment is expected to be dominated by heterogeneous responses.

Adding further structure to the potential treatment responses is another interesting direction. We currently have a discrete notion of network exposure to treatment and control, but one could ask about responses that depend continuously on the *extent* of exposure. As one simple example, we could consider a response that was linear in k, when a vertex had k exposed neighbors. How could we properly take advantage of such structure to get better estimates? Methods for analyzing bias under network exposure condition misspecification would also be a natural addition to the framework.

CHAPTER 7

FUTURE WORK

Many of the largest datasets in computing today are social or behavioral, and this thesis aims to contribute to a new paradigm for understanding and computing with social data. A natural next step from this thesis is to deploy the graph cluster randomization methodology developed in Chapter 6 (using the partitioning algorithms developed in Chapters 4 and 5) to conduct large-scale network experiments featuring careful experimental designs that consider peer effects and other social interactions, to rigorously investigate social decision-making across diverse domains. The challenge of running such experiments can be viewed as one of the grand challenges of the emerging discipline of computational social science.

Second, the study of Facebook's growth in Chapter 2 shows that online engagement can be driven by social networks in surprising ways. As a broad extension of that work, it would be beneficial to understand much more generally how individuals perceive and derive value from network effects, both social and non-social. When are network effects social, and when are they population effects? And when is "stickiness" a network effect and when does it merely reflect a high-friction user experience? How do the roles of stickiness and networks effects vary between online social network loyalty and other forms of customer loyalty?

Third, the algorithms developed in Chapters 4 and 5 can potentially accelerate a much broader class of machine learning tasks beyond the link prediction task (Facebook's "People You May Know" engine) that was demonstrated. For large distributed data sets, if the structure — e.g. a topic model — were known *a priori*, the structure could then be used to greatly improve the computational efficiency of the task itself. As "big data" keeps getting bigger, escaping this Catch-22 scenario will become increasingly important. It would be beneficial to examine how partial or approximate solutions

to large-scale machine learning problems seeking structure can be used to make their own computation more efficient.

The data deluge of online instrumentation and experimentation is providing tremendous opportunities to understand social human behavior. This thesis analyzes the structure of social data from a computational perspective, while also seeking to "close the loop," showing how such structure can be applied to make computation more efficient and inference more accurate, across a wide range of computational domains.

BIBLIOGRAPHY

[1] L. A. Adamic, E. Adar. Friends and neighbors on the web. Social networks 25.3: 211-230, 2003.

[2] L.A. Adamic,, N. Glance. The political blogosphere and the 2004 US election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*. ACM, 2005.

[3] L. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman. Knowledge sharing and Yahoo Answers: everyone knows something. In *WWW*, pages 665–674. ACM, 2008.

[4] K.J. Ahn, S. Guha, and A. McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *PODS*, p. 5–14, 2012.

[5] W. Aiello, F. Chung, L. Lu. A random graph model for massive graphs. In *STOC*. ACM, 2000.

[6] R. Albert, H. Jeong, A.-L. Barabasi. Internet: Diameter of the world-wide web. Nature 401.6749, p. 130-131, 1999.

[7] J.I. Alvarez-Hamelin, L. Dall'Astra, A. Barrat, A. Vespignani. Large scale networks fingerprinting and visualization using the k-core decomposition. *Advances in Neural Information Processing Systems*, 18:41, 2006.

[8] S. Aral, L. Muchnik, A. Sundarajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proc Nat Acad Sci USA* 106:21544, 2009.

[9] P. Aronow and C. Samii. Estimating average causal effects under general interference. *Working Paper*, September 2012.

[10] L. Backstrom, D. Huttenlocher, J. Kleinberg, X. Lan Group formation in large social networks: membership, growth, and evolution. *Proc 12th ACM SIGKDD Intl Conf on Knowledge Discovery and Data Mining* pp 44-54, 2006.

[11] L. Backstrom, E. Sun, C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *WWW*, p. 61–70, 2010.

[12] L. Backstrom and J. Kleinberg. Network bucket testing. In *WWW*, 2011.

[13] L. Backstrom, E. Bakshy, J. Kleinberg, T. Lento, and I. Rosenn. Center of attention: how Facebook users allocate attention across friends. In *ICWSM*, 2011.

[14] L. Backstrom, P. Boldi, M. Rosa, J. Ugander, and S. Vigna. Four Degrees of Separation. In *WebSci*, 2012.

[15] E. Bakshy, I. Rosenn, C. Marlow, L. Adamic. The role of social networks in information diffusion. In *WWW*, pp. 519-528. ACM, 2012.

[16] B. Ball, M. E. J. Newman. Friendship networks and social status. Network Science 1.01, p. 16-30, 2013.

[17] A.-L. Barabsi, R. Albert. Emergence of scaling in random networks. Science 286.5439, p. 509-512, 1999.

[18] M.J. Barber, J.W. Clark. Detecting network communities by propagating labels under constraints. *Physical Review E*, 80:026129, 2009.

[19] V. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J of Statistical Mechanics: Theory and Experiment*, 2008(10):10008, 2008.

[20] M. Berkelaar. The lpsolve package. *http://lpsolve.sourceforge.net*, 2011.

[21] N.E. Biggs, K. Lloyd, R. J. Wilson. Graph Theory 1736-1936. Oxford University Press, 1976.

[22] B. Bollobás. *Random Graphs*, Cambridge Univ Press, 2001.

[23] R.M. Bond, C.J. Fariss, J.J. Jones, A.D.I. Kramer, C. Marlow, J.E. Settle, J.H. Fowler. A 61-million-person experiment in social influence and political mobilization. Nature 489, no. 7415, p. 295-298, 2012.

[24] S. P. Borgatti and M. G. Everett. Models of core/periphery structures. *Social Networks*, 21(4):375–395, 2000.

[25] C. Borgs, J. T. Chayes, L. Lovász, V. Sos, B. Szegedy, and K. Vesztergombi. Counting graph homomorphisms. In *Topics in Discrete Mathematics,* eds. M. Klazar, J. Kratochvil, M. Loebl, J. Matousek, R. Thomas, and P. Valtr, pages 315–371. Springer, 2006.

[26] S.P. Boyd, L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[27] R. Burt. Social contagion and innovation: cohesion versus structural equivalence. *Am J Sociol* 92:1287, 1987.

[28] R. Burt. *Structural holes: the social structure of competition*. Harvard Univ Press, 1992.

[29] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, E. Shir. A model of Internet topology using k-shell decomposition. *Proc Nat Acad Sci USA* 104:11150, 2007.

[30] D. Cartwright, F. Harary. Structural balance: a generalization of Heider's theory. Psychological review 63.5, p. 277, 1956.

[31] D. Cellai, A. Lawlor, K. Dawson, J. Gleeson. Critical phenomena in heterogeneous k-core percolation. *Phys Rev E*, 87(2):022134, 2013.

[32] D. Centola, V. Eguíluz, M. Macy. Cascade dynamics of complex propagation. *Physica A* 374:449, 2007.

[33] D. Centola, M. Macy. Complex contagions and the weakness of long ties. *Am J Sociol* 113:702, 2007.

[34] D. Centola. The spread of behavior in an online social network experiment. Science 329.5996, p. 1194-1197, 2010.

[35] D. Centola. An experimental study of homophily in the adoption of health behavior. Science 334.6060, p. 1269-1272, 2011.

[36] S. Chatterjee and P. Diaconis. Estimating and understanding exponential random graph models. *arXiv preprint*, arXiv:1102.2650v3, 2011.

[37] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan. On compressing social networks. In *KDD*, p. 219–228, 2009.

[38] N.A. Christakis, J.H. Fowler. The spread of obesity in a large social network over 32 years. *New Engl J Med* 357:370, 2007.

[39] N.A. Christakis, J.H. Fowler. Social network sensors for early detection of contagious outbreaks. PLOS one 5, no. 9, e12948, 2010.

[40] J.D. Cohen Trusses: Cohesive Subgraphs for Social Network Analysis. *National Security Agency Technical Report*, 2008.

[41] R. Cohen, S. Havlin, D. Ben-Avraham. Efficient immunization strategies for computer networks and populations. Physical Review Letters 91, no. 24, p. 247901, 2003.

[42] R. Cooley, B. Mobasher, J. Srivastava. Web mining: Information and pattern discovery on the world wide web. In *Proceedings 9th IEEE Conference on Tools with Artificial Intelligence*, 1997.

[43] D. Crandall et al. Inferring social ties from geographic coincidences. *Proc Nat Acad Sci USA* 107:52, 22436, 2010.

[44] J. Davis and S. Leinhardt. The structure of positive interpersonal relations in small groups. In *Sociological Theories in Progress. Vol. 2*, eds. J. Berger, M. Zelditch, and B. Anderson. Houghton-Mifflin, 1971.

[45] P. Dodds, D. Watts. Universal behavior in a generalized model of contagion. *Phys Rev Lett* 92:218701, 2004.

[46] R. Durrett. *Ten Lectures on Particle Systems*, Springer, 1995.

[47] P. Erdős, A. Erdős-Rényi. On random graphs. Publicationes Mathematicae Debrecen 6, p. 290-297, 1959.

[48] L. Euler, Solutio problematis ad geometriam situs pertinentis, Commentarii Academiae Scientiarum Imperialis Petropolitanae 8, p. 128-140, 1736.

[49] M. Faloutsos, P. Faloutsos, C. Faloutsos. On power-law relationships of the internet topology. ACM SIGCOMM Computer Communication Review. Vol. 29. No. 4. ACM, 1999.

[50] K. Faust. Very local structure in social networks. *Sociological Methodology*, 37(1):209–256, 2007.

[51] K. Faust. A puzzle concerning triads in social networks: Graph constraints and the triad census. *Social Networks*, 32(3):221–233, 2010.

[52] U. Feige, R. Krauthgamer. A polylogarithmic approximation of the minimum bisection. In *FOCS*, 105–115, 2000.

[53] S. E. Fienberg. A brief history of statistical models for network analysis and open challenges. *Journal of Computational and Graphical Statistics*, 21(4):825–839, 2012.

[54] D. Fisher, M. A. Smith, and H. T. Welser. You are who you talk to: Detecting roles in usenet newsgroups. In *HICSS*, 2006.

[55] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[56] H.N. Fowler. *Plato: Phaedrus*, Loeb Classical Library, 1971.

[57] J.H. Fowler, N.A. Christakis. Cooperative behavior cascades in human social networks. *Proc Nat Acad Sci USA* 107:5334, 2010.

[58] M.R. Garey, D.S. Johnson, L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical computer science*, 1(3):237–267, 1976.

[59] M. R. Garey and D. S. Johnson. *Computers and intractability*, 1979.

[60] D. Gibson, J. Kleinberg, P. Raghavan. Inferring web communities from link topology. In *Proceedings of the ninth ACM conference on Hypertext and hypermedia*, ACM, 1998.

[61] M. Girvan, M. E. J. Newman. Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99.12, p. 7821-7826, 2002.

[62] B. H. Good, Y.-A. de Montjoye, and A. Clauset. Performance of modularity maximization in practical contexts. *Phys Rev E*, 81:046106, 2010.

[63] M. Granovetter. The strength of weak ties. *Am J Sociol* 78:1360, 1973.

[64] M. Granovetter. Threshold models of collective action. *Am J Sociol* 83:1420, 1978.

[65] S. Gregory. Finding overlapping communities in networks by label propagation *New Journal of Physics*, 12:103018, 2010.

[66] A. Gupta, R. Krauthgamer, J. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *FOCS*, 2003.

[67] F. Harary, R. Z. Norman. Graph theory as a mathematical model in social science. University of Michigan Research Center for Group Dynamics, 1953.

[68] G. C. Homans. *The Human Group*. Harcourt, Brace and World, New York, 1950.

[69] D. Cartwright, F. Harary, R. Z. Norman. *Structural models: An introduction to the theory of directed graphs.* Wiley and Sons, New York,1965.

[70] C. Haythornthwaite, B. Wellman Work, friendship, and media use for information exchange in a networked organization. *J Am Soc Inform Sci* 49:1101, 1998.

[71] B. Hopkins, R. J. Wilson. The Truth about Königsberg. The College Mathematics Journal 35.3, p. 198-207, 2004.

[72] D. Horvitz, D. Thompson. A generalization of sampling without replacement from a finite universe. *JASA*, 1952.

[73] B. A. Huberman, L. A. Adamic. Internet: growth dynamics of the world-wide web. Nature 401.6749, p. 131-131, 1999.

[74] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *PKDD '00*, pages 13–23, 2000.

[75] S. Janson, M.J. Muczak. A simple solution to the k-core problem. *Random Struct Algo* 30:1-2, 50–62, 2007.

[76] E. M. Jin, M. Girvan, and M. E. J. Newman. The structure of growing social networks. *Phys. Rev. E*, 64:046132, 2001.

[77] D. Karger, M. Ruhl. Finding nearest neighbors in growth-restricted metrics. In *STOC*, 2002.

[78] G. Karypis, V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *J. Parallel Distrib. Computing* 48(1):96–129, 1998.

[79] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J Sci Computing*, 20(1):359–392, 1998.

[80] G. Karypis and V. Kumar. Parallel multilevel series k-way partitioning scheme for irregular graphs. *SIAM Review*, 41(2):278–300, 1999.

[81] L. Katzir, E. Liberty, O. Somekh. Framework and algorithms for network bucket testing. In *WWW*, 2012.

[82] M. Kearns, S. Suri, N. Montfort. An experimental study of the coloring problem on human subject networks. *Science* 313:824, 2006.

[83] B. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, 1970.

[84] A. Khan, N. Li, X. Yan, Z. Guan, S. Chakraborty, and S. Tao. Neighborhood based fast graph search in large networks. In *SIGMOD*, pages 901–912, 2011.

[85] B. Klimt, Y. Yang. The Enron Corpus: A new dataset for email classification research. Machine learning: ECML 2004. Springer Berlin Heidelberg, p. 217-226, 2004.

[86] J Kleinberg. Authoritative sources in a hyperlinked environment. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998. Also appears as IBM Research Report RJ 10076, May 1997.

[87] R. Kohavi, A. Deng, B. Frasca, R. Longbotham, T. Walker, Y. Xu. Trustworthy online controlled experiments: five puzzling outcomes explained. In *KDD*, 2012.

[88] D. Kőnig. Theorie der endlichen und unendlichen Graphen, Leipzig: Akademische Verlagsgesellschaft, 1936.

[89] G. Kossinets, D. Watts. Empirical analysis of an evolving social network. *Science* 311:88, 2006

[90] R. Kumar, J. Novak, A. Tomkins. Structure and Evolution of Online Social Networks. KDD, 2006

[91] M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Trans. on Knowledge and Data Engineering*, 16(9), 2004.

[92] A. Kyrola, G. Blelloch, and C. Guestrin. Graphchi: Large-scale graph computation on just a PC. In *OSDI*, 2012.

[93] M. Larsson, J. Ugander. A Concave Regularization Technique for Sparse Mixture Models. Advances in Neural Information Processing Systems (NIPS) 24, 2011.

[94] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD*, pages 177–187. ACM, 2005.

[95] J. Leskovec, E. Horvitz. Planetary-scale views on a large instant-messaging network. In *WWW*, 2008.

[96] J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW*, p. 695–704, 2008.

[97] J. Leskovec, K. Lang, A. Dasgupta, M. Mahoney. Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics* 6(1):29–123, 2009.

[98] G. Li, M. Semerci, B. Yener, and M. Zaki. Effective graph classification based on topological and label attributes. *Statistical Analysis and Data Mining*, 4(5):265–283, 2012.

[99] D. Liben-Nowell, J. Kleinberg. The Link Prediction Problem for Social Networks. In *CIKM*, 2003.

[100] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, A. Tomkins. Geographic routing in social networks. *PNAS*, 102:11623, 2005.

[101] T. Liggett. *Interacting Particle Systems*, Springer, 1985.

[102] X. Liu and T. Murata. Advanced modularity-specialized label propagation algorithm for detecting communities in networks. *Physica A*, 389(7):1493–1500, 2010.

[103] L. Lovász. Very large graphs. In D. Jerison, B. Mazur, T. Mrowka, W. Schmid, R. Stanley, and S. T. Yau, editors, *Current Developments in Mathematics*, pages 67–128. International Press, 2009.

[104] T. Luczak. Size and connectivity of the k-core of a random graph. *Discrete Math* 91:1, 61–68, 1991.

[105] C. Manski. Identification of treatment response with social interactions. *The Econometrics Journal*, 16(1):S1–S23, 2013.

[106] C. Marlow. The structural determinants of media contagion. PhD Thesis, Massachusetts Institute of Technology, 2005.

[107] A. McCallum, A. Corrada-Emmanuel, X. Wang. Topic and role discovery in social networks. In *IJCAI*, 2005.

[108] S. Milgram. The small world problem. Psychology today 2.1 p. 60-67, 1967.

[109] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298:824–827, 2002.

[110] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *IMC*, 2007.

[111] M. Molloy, B. Reed. The Size of the Largest Component of a Random Graph on a Fixed Degree Sequence. Combinatorics, Probability and Computing 7 , 295-306, 1998.

[112] J. W. Moon and L. Moser. On a problem of Turan. *Magyar Tud. Akad. Mat. Kutat Int. Kzl*, 7:283–286, 1962.

[113] J.L. Moreno. *Who Shall Survive? Foundations of sociometry, group psychotherapy and socio-drama* , Nervous and Mental Disease Pub. Co., 1934.

[114] E. Mossel, S. Roch. On the submodularity of influence in social networks. *Proc ACM Symoposium on Theory of Computing*, pp 128, 2007.

[115] P.J. Mucha, T. Richardson, K. Macon, M.A. Porter, and J.P. Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328:876, 2010.

[116] J. Neville, B. Gallagher, T. Eliassi-Rad, and T. Wang. Correcting evaluation bias of relational classifiers with network cross validation. *Knowledge and Info Sys*, p. 1–25, 2012.

[117] M. E. J. Newman, S. H. Strogatz, D. J. Watts. Random graphs with arbitrary degree distributions and their applications Physical Review E 64.2, 026118, 2001.

[118] M. E. J. Newman, D. Watts Random graph models of social networks. *Proc Natl Acad Sci USA* 99:2566, 2002.

[119] M.E.J. Newman, M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, 2004.

[120] M.E.J. Newman. Modularity and community structure in networks. *PNAS*, 103(23): 8577–8582, 2006.

[121] M. E. J. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[122] J. Nishimura, J. Ugander. Restreaming Graph Partitioning: Simple Versatile Algorithms for Advanced Balancing. In *KDD*, 2013.

[123] L. Page, S. Brin, R. Motwani, T. Winograd. The PageRank citation ranking: Bringing order to the web. *Stanford Digital Library Technologies Project TR*, Jan. 1998. (Early version: L. Page. PageRank: Bringing order to the web. Stanford Digital Libraries Working Paper 1997-0072, Stanford University, 1997.)

[124] G. Palla, A. L. Barabási, T. Vicsek. Quantifying social group evolution. *Nature* 446:664, 2007.

[125] R. Pastor-Satorras, A. Vespignani. Epidemic spreading in scale-free networks. *Phys Rev Lett* 86:3200, 2001.

[126] J. Park and M. E. J. Newman. Solution for the properties of a clustered network. *Phys Rev E*, 72(2):26136, 2005.

[127] U.N. Raghavan, R. Albert, S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76:036106, 2007.

[128] A. Rapoport. Spread of information through a population with socio-structural bias I: Assumption of transitivity. *Bulletin of Mathematical Biophysics*, 15(4):523–533, December 1953.

[129] A. Razborov. Flag algebras. *Journal of Symbolic Logic*, 72:1239–1282, 2007.

[130] A. Razborov. On the minimal density of triangles in graphs. *Combinatorics, Probability and Computing*, 17:603–618, 2008.

[131] Resnick, Michael D., et al. Protecting adolescents from harm: findings from the National Longitudinal Study on Adolescent Health. JAMA 278.10, p. 823-832, 1997.

[132] P.R. Rosenbaum, D.B. Rubin. The central role of the propensity score in observational studies for causal effects. Biometrika 70.1 p. 41-55, 1983.

[133] D. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *J. Ed. Psych.*, 1974.

[134] T. Schelling (1971) Dynamic models of segregation. *J Math Sociol* 1:143.

[135] C.R. Shalizi, A.C. Thomas. Homophily and contagion are generically confounded in observational social network studies. *Sociological Methods & Research*, p. 211-239, 2011.

[136] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt. Efficient graphlet kernels for large graph comparison. In *AISTATS*, 2009.

[137] A. Sidorenko. A correlation inequality for bipartite graphs. *Graphs and Combinatorics*, 9:201–204, 1993.

[138] G. Simmel. *Conflict and the web of group affiliations*, (Free Press), 1922.

[139] `http://snap.stanford.edu/data/`.

[140] D. A. Spielman and S.-H. Teng. Spectral partitioning works: planar graphs and finite element meshes. In *FOCS*, p. 96–105, 1996.

[141] H. E. Stanley. Scaling, universality, and renormalization: Three pillars of modern critical phenomena. *Reviews of modern physics*, 71.2, 1999.

[142] I. Stanton, G. Kilot. Streaming graph partitioning for large distributed graphs. *KDD*, 1222–1230, 2012.

[143] I. Stanton. Streaming balanced graph partitioning for random graphs. *arXiv preprint arXiv:1212.1121*, 2012.

[144] D. Strauss. On a general class of models for interaction. *SIAM Review,* 28(4):513–527, 1986.

[145] E. Sun, I. Rosenn, C. Marlow, T. Lento. Gesundheit! modeling contagion through Facebook news feed. *Proc AAAI Intl Conf on Weblogs and Social Media*, pp 146, 2009.

[146] E. Tchetgen, T. VanderWeele. On causal inference in the presence of interference. *Stat. Meth. Med. Res.*, 2012.

[147] P. Toulis, E. Kao. Estimation of Causal Peer Influence Effects. In *ICML*. 2013.

[148] A. Thusoo, JS Sarma, N Jain, Z Shao, P Chakka, N Zhang, S Antony, H Liu, R Murthy. Hive – a petabyte scale data warehouse using hadoop. In *ICDE*, 996–1005, 2010.

[149] C. Tsourakakis, C. Gkantsidis, B. Radunovic, and M. Vojnovic. Fennel: Streaming graph partitioning for massive scale graphs. *Microsoft Technical Report MSR-TR-2012-113*, 2012.

[150] C. Tsourakakis, C. Gkantsidis, B. Radunovic, and M. Vojnovic. Fennel: Streaming graph partitioning for massive scale graphs. In *WSDM*, 2014.

[151] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the Facebook social graph. Technical Report cs.SI/1111.4503, arxiv, November 2011.

[152] J. Ugander, L. Backstrom, C. Marlow, and J. Kleinberg. Structural diversity in social contagion. *PNAS*, 109(16):5962–5966, 2012.

[153] J. Ugander and L. Backstrom. Balanced label propagation for partitioning massive graphs. In *WSDM*, 2013.

[154] J. Ugander, L. Backstrom, J. Kleinberg. Subgraph Frequencies: Mapping the Empirical and Extremal Geography of Large Graph Collections. In *WWW*, 2013.

[155] J. Ugander, B. Karrer, L. Backstrom, and J. Kleinberg. Graph cluster randomization: Network exposure to multiple universes. In *KDD*, 2013.

[156] S. Vishwanathan, N. Schraudolph, R. Kondor, and K. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 99:1201–1242, 2010.

[157] J. Voss. Measuring Wikipedia. In *ICISSI*, 2005.

[158] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge Univ. Press, 1994.

[159] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[160] D. Watts D, P. Dodds. Influentials, networks, and public opinion formation. *J Consumer Res* 34:441, 2007.

[161] X. Yan and J. Han. gpsan: Graph-based substructure pattern mining. In *ICDM '02*, pages 721–724, 2002.

[162] X. Zhu, Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. CMU CALD Tech Report CMU-CALD-02-107, 2002.