ANALYSIS OF LARGE-SCALE DATA FROM HUMAN ACTIVITIES ON THE WEB

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
Lars Backstrom
August 2009
This work focuses on data mining and machine learning using large-scale datasets, with an emphasis on Web information, social computing, and on-line social networks. These datasets are becoming more numerous, and as the Web’s reach grows, it is important to understand these datasets for two reasons. First, better understanding of the systems generating the data allows us to improve the systems. For example, by looking at where search queries come from, we can better select what results and advertisements to display. Second, an in-depth understanding of the data allows us to leverage it for a variety of purposes. For instance, by looking at the geographic sources of queries we can discover the reach of various ideas.

In particular we will develop new algorithms to deal with these large datasets, answering the subtle and nuanced questions that require a huge amount of data and novel methodology. We will examine large social networks, and processes related to these networks such as group formation and network evolution. We will also look at data from web search, showing that it is a rich source of information which, when combined with IP address geolocation can tell us a great deal about the geographic extent of various terms. In addition to learning about these systems, we will also design algorithms for improving them. Through the use of server logs, we will show how changing content can be scheduled more optimally on web pages. Finally, we will examine some of the privacy implications of this style of research, showing a negative result which illustrates how careful we must be with our data.
BIOGRAPHICAL SKETCH

Lars Backstrom was born September 4, 1982 in Chicago. He grew up in Fairfield, Iowa where he became a high-school dropout in 1998. After taking correspondence courses and classes at the University of Iowa, and playing a lot of computer games, he went to Cornell University in 2000, graduating with a B.S. in Computer Science in 2004. In 2005 he became a graduate student in Computer Science at Cornell, where he expects to graduate with a Ph.D. in August of 2009. After Cornell, he will finally get a real job at Facebook.
ACKNOWLEDGEMENTS

First, I’d like to thank my advisor Jon Kleinberg for all his help and collaboration over the last four years. It’s been a great pleasure to work with him. I’d also like to thank my other two committee members, Dan Huttenlocher and Thorsten Joachims, who have given me much beneficial advice. Also, I’d like to thank Rich Caruana without whom I probably wouldn’t have made it into graduate school.

I’d also like to thank all my other coauthors without whom this thesis clearly could not have happened. Jon and Dan have been especially influential and important in shaping the work here and in guiding all of my research. Jure Leskovec has also been a pleasure to work with and has played an important role in my research. Also, I’d like to thank Xiangyang Lan for his important contributions to Chapter 3. Of course, Yahoo! Research has played a huge part in much of my work, providing data, funding and most importantly brilliant collaborators. Ravi Kumar and Andrew Tompkins were especially helpful with their advice and input. I’d also like to thank Jasmine Novak for her great work on our paper which led to Chapter 5, and Cynthia Dwork for her input on privacy.

Finally, I’d like to thank my family for always being supportive.
# TABLE OF CONTENTS

Biographical Sketch ........................................................ iii
Acknowledgements ............................................................ iv
List of Tables ................................................................. viii
List of Figures ................................................................. x

1 **Introduction** .............................................................. 1
   1.1 Social Networks ....................................................... 2
   1.2 Spatial Variation ..................................................... 4
   1.3 Media Scheduling .................................................... 5
   1.4 Privacy ................................................................. 6

2 **Background** ............................................................... 8
   2.1 Preferential Attachment ........................................... 9
       2.1.1 Citations ...................................................... 10
       2.1.2 Graph Formation ............................................. 10
   2.2 Triadic Closure ..................................................... 12
       2.2.1 Clustering Coefficient ..................................... 12
       2.2.2 Empirical Results .......................................... 14
   2.3 Diffusion ............................................................. 15
       2.3.1 Information Spreading ...................................... 16
       2.3.2 Epidemiological Models .................................... 18
   2.4 Open and Closed Network Structure ............................. 19
   2.5 Maximum Likelihood ............................................... 22

3 **Group Formation in Large Social Networks** ..................... 26
   3.1 Analyzing Group Formation and Evolution ..................... 27
   3.2 Community Membership ........................................... 32
       3.2.1 Dependence on number of friends ......................... 33
       3.2.2 A broader range of features ............................... 37
       3.2.3 Results and Discussion .................................... 40
   3.3 Community Growth ................................................ 45
       3.3.1 Results ...................................................... 46
       3.3.2 Discussion of Results ..................................... 48
   3.4 Movement Between Communities ................................ 49
       3.4.1 Time Series and Detected Bursts ......................... 50
       3.4.2 Papers Contributing to Movement Bursts ................. 53
       3.4.3 Alignment between Different Conferences ............... 55
   3.5 Discussion .......................................................... 59
   3.6 Related Work ...................................................... 60
4 Microscopic Evolution of Social Networks 62
4.1 Models of Network Evolution 62
   4.1.1 Evaluation based on likelihood 63
   4.1.2 Data and Model Structure 65
   4.1.3 Our results 65
4.2 Preliminaries 67
4.3 Preferential attachment 68
   4.3.1 Edge attachment by degree 69
   4.3.2 Edges by the age of the node 71
   4.3.3 Bias towards node age and degree 71
4.4 Locality of edge attachment 74
   4.4.1 Triangle-closing models 77
4.5 Node and edge arrival process 80
   4.5.1 Edge initiation 80
   4.5.2 Node arrivals 84
4.6 A network evolution model 85
   4.6.1 Gaps and power law degree distribution 87
   4.6.2 Unfolding network evolution 89
4.7 Discussion 90
4.8 Related Work 91

5 Spatial Variation in Search Engine Queries 93
5.1 Basic properties of the model 95
   5.1.1 Further extensions to the model 96
5.2 Modeling Spatial Variation 97
   5.2.1 Methodology 97
   5.2.2 Model 98
   5.2.3 Algorithm 99
5.3 Assessing the Model 101
   5.3.1 Evaluation 105
   5.3.2 Exponents and Dispersion 109
5.4 Extensions: Temporal Variation and Multiple Centers 110
   5.4.1 Temporal Aspects 110
   5.4.2 Multiple Centers 115
5.5 Enumerating Multiple Queries on a Shared Map 117
   5.5.1 Distinctive Queries for all Locations 117
   5.5.2 Spheres of Influence 118
5.6 Discussion 120
5.7 Related work 121

6 Optimizing Web Traffic via the Media Scheduling Problem 124
6.1 The Media Scheduling Problem 125
   6.1.1 Our Results for Media Scheduling 126
6.2 The Problem Formalized 128
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3 Data and Temporal patterns</td>
<td>129</td>
</tr>
<tr>
<td>6.4 Algorithms</td>
<td>134</td>
</tr>
<tr>
<td>6.4.1 Preliminaries</td>
<td>134</td>
</tr>
<tr>
<td>6.4.2 Our Algorithm</td>
<td>136</td>
</tr>
<tr>
<td>6.4.3 Performance when conditions not met</td>
<td>143</td>
</tr>
<tr>
<td>6.5 Experiments</td>
<td>144</td>
</tr>
<tr>
<td>6.6 Generative models</td>
<td>147</td>
</tr>
<tr>
<td>6.7 Discussion</td>
<td>151</td>
</tr>
<tr>
<td>6.8 Related work</td>
<td>152</td>
</tr>
<tr>
<td>7 Anonymized Social Networks and Structural Steganography</td>
<td>155</td>
</tr>
<tr>
<td>7.1 Attacks on anonymized social networks</td>
<td>157</td>
</tr>
<tr>
<td>7.1.1 The nature of the attacks</td>
<td>159</td>
</tr>
<tr>
<td>7.1.2 Parameters of the active attacks</td>
<td>161</td>
</tr>
<tr>
<td>7.2 The Walk-Based Attack</td>
<td>163</td>
</tr>
<tr>
<td>7.2.1 Description of the Attack</td>
<td>163</td>
</tr>
<tr>
<td>7.2.2 Analysis</td>
<td>168</td>
</tr>
<tr>
<td>7.2.3 Computational Experiments</td>
<td>175</td>
</tr>
<tr>
<td>7.3 The Cut-Based Attack</td>
<td>178</td>
</tr>
<tr>
<td>7.4 Passive Attacks</td>
<td>183</td>
</tr>
<tr>
<td>7.4.1 Computational Experiments</td>
<td>187</td>
</tr>
<tr>
<td>7.5 Discussion</td>
<td>188</td>
</tr>
<tr>
<td>7.6 Related work</td>
<td>189</td>
</tr>
<tr>
<td>8 Conclusions</td>
<td>191</td>
</tr>
<tr>
<td>8.1 Future Work</td>
<td>193</td>
</tr>
<tr>
<td>8.1.1 Experimentation</td>
<td>193</td>
</tr>
<tr>
<td>8.1.2 Multi-Modal Studies</td>
<td>195</td>
</tr>
<tr>
<td>8.1.3 Scalability</td>
<td>195</td>
</tr>
<tr>
<td>Bibliography</td>
<td>197</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Features.</td>
</tr>
<tr>
<td>3.2</td>
<td>Prediction performance for single individuals joining communities in LiveJournal. For every individual in the fringe of one of our 875 communities, we estimate the probability that person will join in a one-month interval. We repeat this experiment using 3 sets of features: only the number of friends in the community, features based on post activity (plus basic features: number of friends and community size), and finally the combination of all the features, including the graph-theoretic ones from Table 3.1.</td>
</tr>
<tr>
<td>3.3</td>
<td>Prediction performance for single individuals joining communities in DBLP. For every triple of a year, a conference, and an author who had not published in the conference, but had coauthored with a conference member, we estimate the probability that the author will publish in the conference’s next meeting.</td>
</tr>
<tr>
<td>3.4</td>
<td>Results for predicting community growth: baselines based on three different features, and performance using all features.</td>
</tr>
<tr>
<td>3.5</td>
<td>Fractions of papers containing hot terms. Papers contributing to a movement burst contain elevated frequencies of currently and expired hot terms, but lower frequencies of future hot terms.</td>
</tr>
<tr>
<td>3.6</td>
<td>Frequency of the four patterns relating movement and topical alignment. $B^+$ (resp. $B^-$) denotes that the burst of $w$ at $B$ follows (resp. precedes) the $B \rightarrow C$ movement burst; and analogously for $C$.</td>
</tr>
<tr>
<td>4.1</td>
<td>Network dataset statistics. $E_b$ is the number of bidirectional edges, $E_u$ is the number of edges in undirected network, $E_\Delta$ is the number of edges that close triangles, $%$ is the fraction of triangle-closing edges, $\rho$ is the densification exponent ($E(t) \propto N(t)^\rho$), and $\kappa$ is the decay exponent ($E_h \propto \exp(-\kappa h)$) of the number of edges $E_h$ closing $h$ hop paths (see Section 4.4 and Figure 4.4).</td>
</tr>
<tr>
<td>4.2</td>
<td>Triangle-closing models. First pick intermediate node $v$ (fix column), then target node $w$ (fix row). The cell gives percent improvement over the log-likelihood of picking a random node two hops away (baseline).</td>
</tr>
<tr>
<td>4.3</td>
<td>Edge gap distribution: percent improvement of the log-likelihood at MLE over the exponential distribution.</td>
</tr>
<tr>
<td>4.4</td>
<td>Node arrival functions.</td>
</tr>
<tr>
<td>4.5</td>
<td>Predicted by Theorem 4.6.1 vs true degree exponents.</td>
</tr>
<tr>
<td>5.1</td>
<td>Accuracy of algorithms for localizing senators inside their respective states.</td>
</tr>
<tr>
<td>5.2</td>
<td>Estimation of exponents $\alpha$ for high-circulation U.S. newspapers.</td>
</tr>
<tr>
<td>5.3</td>
<td>Estimation of exponents $\alpha$ for the 10 most populous U.S. cities.</td>
</tr>
</tbody>
</table>
5.4 Estimation of exponents $\alpha$ for the 10 highest-ranked U.S. universities according to U.S. News & World Report. . . . . . . . . . . . . 109

6.1 The percent improvement of our algorithm over the actual data, and over the simpler baseline algorithm. On all days, our algorithm is within 0.1% of optimal. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 144
# LIST OF FIGURES

2.1 The distribution of citations counts follows a power-law for reasonably large citation counts, as predicted by [106]. ........................................ 11

2.2 Fitting a rough power-law distribution. The true distribution is $\frac{4}{x(x+1)(x+2)}$. ................................................................. 25

3.1 The probability $p$ of joining a LiveJournal community as a function of the number of friends $k$ already in the community. Error bars represent two standard errors. ........................................... 31

3.2 The probability $p$ of joining a DBLP community as a function of the number of friends $k$ already in the community. Error bars represent two standard errors. .................................................... 32

3.3 The top two levels of decision tree splits for predicting single individuals joining communities in LiveJournal. The overall rate of joining is 8.48e-4. ................................................................. 41

3.4 The probability of joining a LiveJournal community as a function of the internal connectedness of friends already in the community. Error bars represent two standard errors. ........................................... 42

3.5 The top two levels of decision tree splits for predicting community growth in LiveJournal. ................................................................. 46

3.6 The rate of community growth as a function of the ratio of closed to open triads: having a large density of closed triads (triangles) is negatively related to growth. Error bars represent two standard errors. .................................................... 47

3.7 Four patterns of author movement and topical alignment: in each of (a)-(d), the labeled arrows represent term burst intervals for a shared hot term in conferences $B$ and $C$, and the square wave represents a $B \rightarrow C$ movement burst. In the terminology from the text, (a) is shared interest, (b) is colonization, (c) is exploration, and (d) is shared membership. ................................................ 56

3.8 Projections of a subset of the conferences in three different years (1980, 1987, 2003) into a two-dimensional LSI space. ........................................ 58

4.1 Probability $p_e(d)$ of a new edge $e$ choosing a destination at a node of degree $d$. ................................................................. 69

4.2 Average number of edges created by a node of age $a$. .................. 70

4.3 Log-likelihood of an edge selecting its source and destination node. Arrows denote $\tau$ at highest likelihood. ................................. 72

4.4 Number of edges $E_h$ created to nodes $h$ hops away. $h = 0$ counts the number of edges that connected previously disconnected components. .... 74

4.5 Probability of linking to a random node at $h$ hops from source node. Value at $h = 0$ hops is for edges that connect previously disconnected components. ........................................ 75
4.6 Triangle-closing model: node $u$ creates an edge by selecting intermediate node $v$, which then selects target node $w$ to which the edge $(u, w)$ is created. .................................................. 77
4.7 Exponentially distributed node lifetimes. ........................................... 81
4.8 Edge gap distribution for a node to obtain the second edge, $\delta(1)$, and MLE power law with exponential cutoff fits. ........................... 83
4.9 Evolution of the $\alpha$ and $\beta$ parameters with the current node degree $d$. $\alpha$ remains constant, and $\beta$ linearly increases. ....................... 84
4.10 Number of nodes over time. ................................................................. 85
4.11 We take FlickR network at first half of its evolution. Then we simulate the evolution using our model and PA for the second half, and compare the obtained networks with the real FlickR network. Notice our model matches the macroscopic statistical properties of the true FlickR network very well, and in fact much better than PA. .................................................. 90

5.1 Geolocation of queries “Red Sox,” “Grand Canyon National Park,” and “Bell South”. (The capitalization of queries is reduced to a canonical form in our experiments.) These figures are drawn as heat maps, with the color spectrum indicating the query intensity per grid cell (and hence there is value in viewing these, as well as later figures, on a color display or color print-out). The arrows indicate the centers computed using our model. .................... 102
5.2 Performance of algorithms on baseball team queries. ......................... 105
5.3 Performance of algorithms on high-population U.S. cities. ................. 106
5.4 The path of Hurricane Dean’s storm center, moving west through the Caribbean, alongside the smoothed path of query centers for “Hurricane Dean.” .................................................. 111
5.5 Change in the exponent for “Hurricane Dean” by hour, as interest in the topic shifted from local to national. ................................. 112
5.6 Multiple centers for the query “United Airlines.” ............................ 113
5.7 Distinctive queries for locations in the San Francisco Bay Area. .. 114
5.8 Spheres of influence of baseball teams. .............................................. 115
5.9 Automatically generated landmark map of New York City. The maps itself simply shows the geotagged image density. ................. 122
5.10 Automatically generated landmark map of London. The map itself simply shows the geotagged image density. ......................... 123

6.1 A screenshot of the Yahoo! front page showing the placement of the featured article: “Best and worst French fries.” ...................... 129
6.2 The number of page views of yahoo.com over the course of three consecutive weeks in Nov/Dec 2008. Note, that with the exception of a small glitch (likely cause by a logging error or server failure) the traffic is very consistent over the three weeks. .................. 130
6.3 An alignment of all clickthrough rate curves to a single universal curve. The red region of the plot shows that almost all of the data fits to within a 10% window around a single curve.

6.4 A fit of three functions to the single longest running article in our dataset. The exponential decay curve is clearly the best, and while the linear curve isn’t too far enough, it will certainly continue to deviate more and more from reality since it will eventually become negative.

6.5 Illustration of the proof of Lemma 6.4.3. If we have a situation where the ‘better’ (blue) item is placed earlier, we can swap the two items as illustrated here. After this swap, the clickthrough rate remains unchanged in the last $\gamma$, while it drops in the lower traffic region from 0 to $\delta$, and rises in the higher traffic region from $\delta$ to $\delta + \sigma_j$.

6.6 This figure shows our scheduling and the optimal scheduling for one particular day. Because the page views are not quite unimodal, our ordering is not quite optimal. It is close enough that the fraction lost is only 3.3E-5 though. For comparison, our baseline method does 2.2% worse, and we estimate a 26% improvement over the ordering that was used.

6.7 The distribution of visit rates to yahoo.com.

6.8 The distribution of interarrival gaps.

6.9 The clickthrough rate declines as visitors repeated see the same featured item.

6.10 Here we put together the visit rate distribution, interarrival gap distribution, and probability of clicking an article as a function of times seen on home page. The figure above shows the actual clickthrough rates along with simulated clickthrough rates for one article and a corresponding set of model parameters.

7.1 For two different choices of $d_0$ and $d_1$, the value $k = 7$ gives the attack on the LiveJournal graph a high probability of success. Both of these choices for $d_0$ and $d_1$ fall well within the degrees typically found in $G$.

7.2 Probability of success for different coalition sizes, in the LiveJournal graph. When only the degrees and internal structure of the coalition are taken into account, a coalition of size 5 is needed to give a high probability of success. When the more refined version of the algorithm is used, and the edges connecting $H$ to $G - H$ are considered, only 4 users need collude.
7.3 As the size of the coalition increases, the number of users in the LiveJournal graph compromised under the passive attack when the coalition successfully finds itself increases superlinearly. The number of users the semi-passive attack compromises increases exponentially.
CHAPTER 1
INTRODUCTION

The Web continues to grow and evolve, playing an ever-increasing role in people’s lives. Huge datasets of human behavior are created from the Web’s usage, which give scientists an opportunity to both understand the Web systems being used, and also perhaps to gain insights into human behavior. From a practical point of view, there are two broad reasons to study these datasets. First, better understanding of the dynamics of the systems generating the data allows us to improve the systems. For example, by looking at the geographic sources of search queries, we can better select what results and advertisements to display. Second, an in-depth understanding of the data allows us to leverage it for a variety of purposes. Looking at where search queries come from we can discover the reach of various topics, ideas, and opinions; and by understanding how new ideas spread on a social network, we can improve marketing efficiency.

These datasets come in a variety of forms and can be analyzed along a number of dimensions. In this work, we will examine a number of different datasets breaking down our analysis into three categories. First, in many datasets, relationships exist between various entities and hence the data can be best viewed as a network. The most obvious instance of these is in the analysis of social networks, though other data can also be viewed in this way: citation networks and group membership (as a bipartite graph) to name two. The second dimension that plays an important role in much of this analysis is the temporal dimension. Many of the interesting features of these data grow and evolve over time, and understanding the mechanisms by which this happens is an important part of understanding the systems generating the data. Finally, many of these datasets have a geographic component to them.
In the case of networks, the nodes can be annotated with location. In the case of search logs, the queries can be annotated by the location of the querier, and in general we can annotate most data by recording the location from which it was generated.

1.1 Social Networks

Social networks have always existed implicitly, but only recently have we been able to study explicit networks on a large-scale. While researchers of the last century made some theoretical progress and were able to do small-scale empirical studies, the advent and popularity of social networking websites allows theoretical models to be tested, and lets us examine real networks on a scale not previously possible. Most analysis of social networks falls into one of two categories. Sociologists typically studied networks in great depth, but were limited to small-scale networks because the methods they used involved questionnaires and interviews and hence did not scale well to thousands of people. On the other hand, in computer science the networks being analyzed were typically much larger, but the questions being asked were consequently much simpler. The focus was on simple network properties like degree distribution and diameter – interesting properties, but somewhat limited.

In Chapter 3 we try to bridge this gap by using techniques from computer science, and in particular data mining and machine learning, to ask more complex questions and tease apart subtle distinctions, all on large-scale datasets which have only become available recently. In particular, we look in depth at questions related to groups or communities of individuals from two datasets. Over time,
these groups grow and evolve, and our goal is to tie network properties to this evolution. Using techniques from machine learning, we are able to discover which features of the social network are most important in predicting whether a person will join a group or not. We find, for instance, that the most important factor is how many friends one has already in a group, and we are able to quantify the nature of the dependence of joining probability on number of friends (it shows a diminishing returns shape). At a finer level of detail, we find that if one’s friends are well connected, that also increases one’s chance of joining: a person with three friends who are all friends with each other is more likely to join than a person with three independent friends. In addition to looking at individuals, we also look at groups as a whole and examine how the network structure of the group is related to its growth rate. Here we find that to accurately predict group growth, one must take into account a broad range of network features and that, while no one network feature is an accurate predictor on its own, when conjoined they are components in an accurate model for predicting the growth of communities. By analysis of these large-scale networks, we are able to answer some questions which have been discussed and theorized about for many years.

In Chapter 4 we will look at the evolution of these networks from another perspective. Instead of examining the spread and growth of groups, we will look at the formation of new edges. Using temporal data, we will examine how these networks evolve at a very fine level of detail. By examining the growth of a number of different networks, we are able to draw some general conclusions about the evolution of social networks, particularly with regards to the appearance of new edges. Using temporally annotated networks, we can evaluate a number of different network evolution models, reporting which ones fit the data most accurately. One of our primary findings here is that most new edges are local; they tend to connect
nodes which were already close to each other in some sense, such as individuals who already had a mutual friend.

1.2 Spatial Variation

Another source of interesting data comes from search engine query logs. While an individual query contains little or no useful information, the entire corpus of billions of queries contains a wealth of information. The changes in the distribution of search terms over time show trends in society, while events appear as sudden spikes in searches for certain terms. A simple example is that one can easily tell when the moon was last full by looking for a day when the query ‘full moon’ had particularly high volume. Another recent example where this information was demonstrated to worldwide fan-fare is Google Flu Trends, where query volume was combined with geolocation to report locations with high flu rates.

In Chapter 5 we will examine what other information can be gained from combining query data with location. We will find that, despite the inaccuracy of geolocation from IP-address, we are able to learn a great deal from the queries and their approximate locations. At the simplest level, we are able to automatically discover the locations of most landmarks and large (but local) organizations. For instance, using this data with the algorithms we develop, we can correctly identify the cities of origin for all major newspapers, sports teams, and many national parks. Furthermore, we discover a number of geographic relationships which are less obvious, such as which social networks are popular in which parts of the world (Facebook was particularly popular in Ontario at the time of this study, for instance). By looking in more depth, we are able to track queries through both time
and space, learning how their geographic profile changes over time. A nice example of this comes from the query ‘Hurricane Dean’, where we are able to recover an approximation of the hurricane’s path by looking only at where the related queries originated.

1.3 Media Scheduling

Server logs which record web traffic could be thought of as a bipartite graph between individuals and web pages. While this view of the data might be useful in some contexts, for other analysis it is simpler to aggregate all of the people together, and examine only the temporal dimension, ignoring the network information. In Chapter 6 we look at server logs to examine the clickthrough rates of various featured news articles. In this case, we don’t care so much who is responsible for the clickthroughs, but are more interested in the aggregate rates.

In particular, we record the traffic to a webpage (yahoo.com) with ever-changing content on it. Over time, the content changes, as does the number of visitors (due to diurnal effects). We capture the rates at which users consume the content (click on it) and examine how the clickthrough rates of various items change over time. From this temporal data, we introduce the media scheduling problem: given the varying visitors rate, the varying content quality, and the varying clickthrough rates schedule content to maximize consumption.

While the problem is NP-Hard in general, we observe that in practice many of the parameters to this problem have some special structure that we can use in the design of our scheduling algorithm. For example, the total number of visitors over the course of a day is roughly unimodal, peaking around midday. Using this,
and other observations from the data, we introduce an efficient algorithm to solve
the variant of the problem which occurs in the data. Our algorithm provides an
optimal schedule, provided these conditions are met, approximating an optimal
solution when the conditions are only approximately met.

1.4 Privacy

In all of this work, we use user-generated data to learn about the world and about
social behavior in particular. In these studies and other related work, we are not
interested in any of the individuals who generated this data, but only in their
aggregate behavior. In some cases, however, the data we use contains sensitive
information. For instance, search query logs may allow users to be identified and
their query histories revealed. Social networks may contain sensitive information
about who communicates with whom. In all of these cases, it is best to err on
the side of caution, and take steps to ensure that none of the participants in the
system being studied have cause for personal privacy concerns in connection with
the study. This presents a dilemma to researchers who would like to have widely
available datasets so that multiple groups can work with the same data, confirming
each other’s results and comparing statistics on the same problems.

In Chapter 7 we will show that even when the data being released seems fairly
innocuous, there are ways for insidious attacks to be performed against the privacy
of the individuals involved, casting doubt on the ability of system curators to widely
release data. Specifically, we show that even when the data is simply an unlabeled,
undirected social network with no other contextual information, an attacker could
compromise the privacy of the individuals in the network. We also show that not
only could an insidious attacker target specific individuals, but that regular users of the system, simply by their normal behavior, could compromise the privacy of the other individuals should the desire strike them to do so.

To carry out these attacks, the attackers need to do no more than create a small number of new nodes in the network, and a small number of edges incident to those new nodes. By doing so, an attacker is able to attack any individual in the system. The attacker does this by connecting the nodes created in a way that is provably unique within the context of the larger network (with high probability). The attacker can then find these new nodes using an algorithm we present. Once the new nodes are found, the attacker can learn the identities of other nodes connected to them.
CHAPTER 2
BACKGROUND

There is a long history of studying networks in many different contexts. In this chapter, we will review some of the work related to the modelling and understanding of networks.

We will start by discussing the preferential attachment model of network evolution. This model gives rise to some macro-level features of real networks, and has been widely studied over the last ten years. The most notable of these features is the presence of a power-law degree distribution. However, despite its success at capturing this feature, it fails to capture others such as locality of new edges, as observed in a number of studies, including Chapter 5.

In addition to the evolution of networks, there is a long history of literature about diffusion on networks. These range from studies of the spread of diseases in epidemiology, to classical studies of new technology adoption, and to more recent studies of mathematical models for diffusion. In all of these cases, there is some set of individuals who are ‘infected’, and they spread the disease or technology along the network to those who are not infected. The details of this process naturally vary depending on the thing which is spreading.

Another important direction of network research comes from sociology, where various local network properties are related to variables like influence and power. For instance, individuals who bridge gaps between disjoint communities may be uniquely situated to spread ideas from one group to another, imparting special powers to them. On the other hand, individuals who are well-connected within their group may tend to have more influence because they are seen as experts in a
single realm.

Evaluating all of these ideas is difficult. In many cases, sound ideas cannot be easily expressed in precise, mathematical terms. In some cases, however, the ideas can be precisely formulated as probabilistic models. For instance, a preferential attachment model of network growth posits precise probabilities for each new edge in the network. These probabilistic models allow us to use the maximum likelihood principle (Section 2.5) which we will apply, particularly in Chapter 4 and Chapter 7. If we have multiple models to explain some phenomena, this framework allows us to say, in a quantitative way, that one model is better than another.

2.1 Preferential Attachment

Preferential attachment refers to the stochastic process whereby the ‘rich get richer’. In general terms, there are a number of entities, each with some number of goods. For example, the entities might be publications and the goods might be citations. New citations (goods) arrive stochastically, but are more likely to go to the publications (entities) that currently have more citations. This concept has been around for many years, first being analyzed in 1925 [134]. An important property of this scheme is that it tends to give rise to power-law distributions. That is, the probability that publication $i$ has $k$ citations after some fixed amount of time is proportional to $k^{-\alpha}$, for some choice of $\alpha$ dependent on the details of the process. Thus, this sort of process has been posited to explain the distribution of wealth, city sizes, and citation count. In this section, we will start with an analysis of the citation case, and then examine a network evolution model based on a similar process.
2.1.1 Citations

We start with a simple model of citations [106] where we imagine that new papers arrive sequentially, and that whenever a new paper arrives, it adds some number $R$ of citations to previous papers. The citations to previous papers are selected preferentially, with each of the new $R$ citations pointing to a paper selected with probability proportional to its current citation count. An important point is that papers must somehow receive some initial weight, or else they would always have a citation count of 0. This is typically done by considering a paper to have an implicit self-citation.

Early analysis of this model [106] showed that in the limiting case of large populations, and reasonably large citation counts, the distribution of citations follows a power-law distribution, with some exponent $\alpha$ greater than 2. In the case where the number of citations added by each new paper is fairly large, this model approaches a continuous variant, which can be easily analyzed and has a power of 2. On the other hand, if each new paper only adds one citation, we approach a power of 3. As we will now see, this matches the power found for a related model of network formation.

2.1.2 Graph Formation

We can easily transform this citation model into a graph formation model. In essence, all that we need to do is keep track of the citation links. When a new paper arrives and a citation is made, we record this as a new vertex and a new edge between citer and citee. This model has been suggested to help explain the scale-free properties observed in many networks [8].
Figure 2.1: The distribution of citations counts follows a power-law for reasonably large citation counts, as predicted by [106].

Since the introduction of this graph formation model, many properties of the graphs formed have been analyzed, both analytically and empirically. Barabasi suggested that the degree distribution would likely have exponent 3, and rigorous analysis has since been shown that the exact distribution will be $p_d \propto \frac{1}{d(d+1)(d+2)}$ [19]. As expected, this exactly matches the citation result for the case where each paper adds only one new citation (as illustrated by the simulation in Figure 2.1). An important difference between the citation model and the graph model is that if a node adds multiple edges when it first arrives, this does not change the distribution, as shown in [19]. By contrast, in the citation model adding more citations does change the distribution. The difference stems from the fact that adding an edge to a network increases the degree of both the source and destination of the edge, while adding a citation increases the citation count only of the cited work.

In addition to the degree distribution of these networks, a number of other
properties have been studied. The average path length between nodes has been shown to be $O(\log N / \log \log N)$ [20]. Other properties have also been examined, such as in and out degree correlations [77] and clustering coefficients [7].

2.2 Triadic Closure

One important feature in the evolution of real social networks is triadic closure – the tendency for edge $(u, w)$ to form when $(u, v)$ and $(v, w)$ are present. This process has been discussed in various forms for many years and its importance in the evolution of networks is well-known [62].

Using longitudinal data [75] this process can be observed directly. In static snapshots of networks, we can view the results of this process by observing a much higher incidence of closed triads (cliques of size 3) than simple random graph models would predict. This observation has given rise to the macro-level graph property known as the clustering coefficient or transitivity of a graph.

2.2.1 Clustering Coefficient

The clustering coefficient [130] can be defined in two ways: a global one and a local one. Both of them rely on the concept of open and closed triads. In both cases, there are edges $(u, v)$ and $(v, w)$. In an open triad, the edge $(u, w)$ is absent, while in a closed triad it is present. To compute the global clustering coefficient, we simply count the number of closed triads, and divide it by the total number of triads. Care must be taken here when deciding whether to count closed triads one time or three (for the three permutations). Typically, closed triads are counted
three times, in both the numerator and denominator of the ratio.

On the other hand, the local clustering coefficient [130] [100] is relative to only a single node. It simply evaluates the fraction of pairs of a node's neighbors who are connected themselves. For instance, if a node has four neighbors, there are 6 possible pairs of them. If two of those pairs are connected, that node's clustering coefficient is $\frac{1}{3}$. An alternative definition (and indeed the original one [130]) of the global clustering coefficient is that it is the average of the local clustering coefficients of all the nodes.

In completely random graph models, where each nodes has fixed degree and is connected to other nodes at random, the clustering coefficient is $k/N$ in expectation, where $k$ is the degree of each node and $N$ is the number of nodes. This is easy to show by considering each triplet $(u,v,w)$, where the probability that $(u,w)$ is an edge, conditioned on $(u,w)$ and $(v,w)$ being edges is simply $k/N$. (A full proof would depend on the details of how such a random graph was generated, since efficiently generating simple graphs with fixed degree uniformly is an open question). For the more complicated preferential attachment random graph model discussed above, the clustering coefficient also decreases as $N$ grows, though more slowly – $C \propto \frac{\log^2 N}{N}$ [57].

These results are in conflict with real networks, which typically have much higher clustering, and whose clustering coefficient typically does not change as the networks grow. For instance, a networks of 225,226 film actors has a clustering coefficient of 0.79 [130], much higher than a random graph of this size would have. This suggests that, while the preferential attachment model is able to roughly capture the degree distribution, it cannot explain other important macro-level graph properties. In reality, edges tend to be much more local than the random
connections which appear in this model.

Regular mesh graphs, on the other hand, do exhibit the high degree of clustering found in real graphs, but of course they do not share other important properties. In particular, they lack the correct degree distribution, and they lack the 'small-world' property where short paths exist between all pairs of nodes. Various models have sought to correct for these shortcomings, achieving various macro-level properties [74] [73] [130].

### 2.2.2 Empirical Results

With the increasing availability of network data, a number of studies have examined the role of triadic closure in the evolution of networks. In an email network studied by Kossinets and Watts [75], it was found that individuals who had a mutual friend (that is, were separated by two hops) were 30 times more likely to become friends than individuals who were separated by three hops.

Given that triads are so much more likely to close than other edges are to form, to accurately reproduce real social networks a random graph model should somehow have this locality property. While the exact degree of triadic closure varies, it seems that in almost all cases, it is much higher than most simple graph models would predict. Also, because these edges seem to play such an important role in graph evolution, it has been useful to examine this process in further detail to understand what other factors there may be. In [75] this process was examined along two dimensions. First, looking only at the network, it was found that the stronger the connections are, the more likely a triad was to be closed. In addition, as the number of mutual acquaintances increased, the probability of
two individuals becoming friends increased significantly. These results all support
the general intuition that as two nodes acquire more and stronger connections to
mutual acquaintances, they become more likely to interact themselves.

In addition to strictly network effects, other factors also can have important
effects on the probability of nodes to become connected. The general principle of
homophily suggests that individuals with similar interests will be more likely to
become friends. Another significant factor may be shared foci – individuals who
share activities are more likely to meet.

In the email network study, the individuals were mostly students, and their
class registrations were known, as well as a number of demographic traits. In this
study, Kossinets and Watts found that the two most important factors in increasing
edge formation probability came from network effects (mutual acquaintances) and
shared foci (individuals in the same classes). Demographic similarities played a
tertiary role, suggesting that homophily tends to show up mostly through shared
foci, rather than being directly responsible for new edges.

2.3 Diffusion

In addition to the processes controlling the growth of the network, other processes
occur on the network which do not effect the network structure, but instead use
it. One widely studied such process is typically known as diffusion – the process
by which something spreads across a network. What the something is can vary
greatly, and different things spread in different ways. For instance, the way in
which a disease spreads may be quite different from the way a rumor spreads.
2.3.1 Information Spreading

The spread and adoption of new ideas has been studied in various contexts for many years. A number of classical studies in the last century examined how new innovations spread across relatively small social networks. For instance, Ryan and Gross [110] studied the adoption of new seed corn by groups of farmers, and found that while many farmers were aware of the new technology from shortly after its invention, some adopted quickly while others took longer. While they did not look directly at the social network, they did notice that more ‘cosmopolitan’ farmers who visited cities more often tended to adopt more quickly. In a later study [34] the social network effects were measured more directly, as it was observed that doctors made many decisions about what technologies to use based on discussions with their colleagues.

These types of processes can be modelled in different ways, some of which are more appropriate than others for different contexts. In the cases discussed above, we can model the process as one where individuals must make some decision based on the imperfect information available to them. We imagine that to make a decision, an individual takes into account both his private, personal information and the information inferred from the actions of his social contacts. Thus, we imagine that in the example of farmers, only a small number of farmers originally did sufficient research and were of the right disposition to adopt the new technology early on. However, a farmer who knew one of these early adopters would observe increased yields, and hear about the advantages from his neighbor. At some point, the weight of one’s contacts’ actions may overwhelm one’s own information and biases, at which point an individual adopts.

To make these ideas concrete, we can imagine that every individual $u$ in a
network has some personal threshold $p_u$, and that once at least a fraction $p_u$ of an individual’s neighbors have adopted a new technology, then that individual will become convinced and adopt himself. To start things off, some individuals have $p_u = 0$ and these individuals adopt the new technology without any convincing from their neighbors. From this point, the technology spreads across the network, eventually reaching some maximum extent at which point no more individuals will adopt the technology. It can easily be shown that, in the simplest version of this model, the order in which individuals are considered is unimportant, and thus the diffusion will reach some maximum set of affected individuals, which is a function only of the network parameters. We can also extend this model in a number of obvious ways, giving the edges weights, or adding time varying parameters.

There are a number of different analyses and results for this model. One observation [30] is that it is difficult for new innovations to spread long distances quickly in many cases. If most of the adoption is concentrated in New York, it is unlikely that individuals in Los Angeles will have enough New York friends who have adopted to put them over the threshold. Consequently, while these long ties can greatly increase the speed of diffusion when the adoption threshold is low, they have less impact as the threshold increases.

Other studies of this model have focused on how to target specific high-impact individuals to maximize adoption. As an advertising company, or a public service agency, it would be greatly beneficial if we could target our message to the individuals who are most important for spreading the new technology. While studies of real networks have been hard to come by, theoretical work [70] suggests that this goal is difficult to achieve for this model.

We can extend this model slightly to a game theory view, where there are two
strategies: adoption and non-adoption [97]. In this view, an individual gains utility
from matching the strategy of his neighbors. For each neighbor who he matches,
he receives some utility if they are both adopters, and some other utility if they
are both non-adopters. This version is known as the coordination game, since
the payoff is related to how well an individual can coordinate with his neighbors.
If an individual gets a payoff \(a\) from coordinating on the adoption strategy and a
payoff of \(b\) from coordinateing on the non-adoption strategy, then it is easy to show
that adoption is a better strategy if the fraction of neighbors who have adopted
is greater than \(\frac{b}{a+b}\). The only difference between this game and the model where
individuals adopt after some threshold is that we now allow individuals to switch
back and forth between adoption and non-adoption.

### 2.3.2 Epidemiological Models

A somewhat different model describes the spread of diseases probabilistically.
While individuals behaved deterministically in the previous models, here we imag-
ine that they are ‘infected’ with some probability each time they are exposed. This
model clearly applies better to infectious diseases than the previous one, and may
be more widely applicable.

More formally, we imagine that there is some probability \(p\), and each time an
individual is infected, he spreads the disease to each of his uninfected neighbors
with probability \(p\). As in the threshold model, the structure of the network plays
a significant role in the extent to which the disease will spread. In contrast to the
threshold model, it is easy to see that in a network of people, long-range links will
play a large role in helping a disease spread. For instance, if no one has contact
with individuals more than a few miles away, the disease can only spread slowly, a
few miles at a time. A few long-range links, however, have the potential to spread
the disease great distances.

Another significant difference between the two models is that while it was hard
to predict the outcome of an infection ahead of time in the threshold model, without
actually running the simulation, this probabilistic model is better behaved in some
sense. In the threshold model, a ‘tipping point’ could be reached where a single
individual’s adoption played a huge role in the final state of the network. Here,
however, things are more continuous, and while a single individual may still play a
large role, this ‘tipping point’ effect is somewhat mitigated [70]. This allows us to
devise (approximate) algorithms which identify which individuals to target if we
want to maximize the extent of diffusion.

An extension to this model allows for individuals to recover from their infec-
tions, granting them temporary immunity which fades over time. The flu, for
instance, can be thought of in this way. Individuals catch the flu, spread it to
some of their contacts, recover and are then insusceptible for some time, after
which they may catch the flu again. An interesting consequence of this is that os-
cillations can occur where the general population goes through periods of increased
immunity, followed by large-scale epidemics due to the simultaneous recovery of
many individuals. As the network becomes more local, with fewer long-range links,
this periodicity disappears on the global scale [83].

\section{Open and Closed Network Structure}

In the various processes that occur on social networks, different nodes are affected
differently or to different degrees. This brings us to the idea of social capital, the
notion that due to their placement in the social network, certain individuals have some sort of advantage over others. There are many different advantages one’s placement in a network might confer. At a simple level, a node with higher degree has better access to information, and more ability to influence people. At a more complex level, we can look not only at the degree of a node, but at its placement in the network. One high-level distinction which has been examined in a long line of social science literature is the difference between nodes placed in the middle of tightly knit communities, versus nodes placed at the intersection of multiple communities. Using the terminology from above – the distinction between nodes with high clustering coefficient and those with low clustering coefficient.

In different ways, both high and low clustering can confer some social advantage to the individuals. It has been argued [33] [32] that mutual friends increase the reliability of information, and increase trust. If $u$ and $v$ are connected and have many mutual friends, they are less likely to cheat each other because all of their mutual friends would observe this, and ostracize whoever was responsible. Thus, being well-connected enables an individual to trust his contacts, removing the cost of verifying information, or setting up legal contracts to enforce agreements. On the other hand, being at the juncture of multiple communities can give significant advantages [26]. Someone with contacts in many disjoint groups receives information from all of those groups. In a research setting, for instance, this enables one to apply techniques from one field to another, potentially leading to significant breakthroughs.

If an individual is in a well-connected dense part of the network, he has many redundant sources of information. He can trust the other individuals whom he knows because they are all known to his other contacts. In addition, an individ-
ual in this structure is more able to repel any sort of attacks against his group. Should an individual violate the norms of the group, the group of well-connected individuals can easily band together. Because they all know each other, a group of well-connected individuals can operate as a team, gaining what powers that confers.

On the other hand, being in the center of a well-connected subgraph has some potential disadvantages. Such an individual has less opportunity to gain access to new information, as his contacts will likely all have much of the same information as he has. He is often unable to ask a friend a question as his friends are similar to him and are no more likely to provide an answer. Furthermore, any answers he might receive are correlated with one another, and thus he gains less information from his contacts than he would from a group of random people.

In these senses, an individual whose contacts span multiple groups has some advantage. This individual fills a 'structural hole' [26]. As a result, he has access to diverse sources of information. Because of his contacts in other groups, he is likely to receive more independent information, and can perhaps make better-informed decisions. Not only does he have an advantage in terms of getting more diverse information, but such an individual acts as an important conduit of information. For information to spread from one group to another, it must pass through such an individual. This gives such an individual an opportunity to control the information spreading between groups, and also to gain some brokerage benefits.

As we see here, social capital can exist in a number of ways. While there is some broad agreement that such a thing exists, there are different views upon what network structures confer the most advantages. There has been some empirical work [27] supporting the 'structural holes' view, though in general it has proved
difficult to do large-scale studies which conclusively support one view or the other.

2.5 Maximum Likelihood

Though it is not specific to networks, one tool which we will make use of in this thesis is the idea of a maximum likelihood estimator. When talking about various models for things like graph evolution, this framework gives us a way to put very different models on equal ground in some sense. It gives us a single quantitative evaluation of various models.

To use this evaluation measure, we think of some process as a sequence of probabilistic events. We imagine that each time we observe an event, it was sampled from some distribution of possible events. Given some model for this distribution of events, it is typically easy to calculate the probability of the events. Our model will tell us the probability distribution \( p_i(\cdot) \) for all possible events at time \( i \), given the state of the system. If the probability of event \( e_i \) according to the model is \( p_i(e_i) \), then the probability of a sequence of events is simply the product \( \prod p_i(e_e) \). We are more interested in the probability of the model given the events, than the other way around. However, these are closely linked by Bayes’ law. We can write

\[
P(model|events) = \frac{P(events|model)P(model)}{P(events)}
\]

The term in the denominator is a constant independent of the model, and hence is irrelevent for finding the maximum likelihood model. The prior probability of the model \( P(model) \), however, is important. Qualitatively, it often lends preference to simpler models but it is usually difficult to evaluate quantitatively.

Given models with the same space of parameters, it is simplest to simply treat
the prior as uniform over all possible choices, for lack of anything better to do with it. In this case, we can simply find the model which maximizes the event probability, and this will be the same as finding the maximum likelihood model. When models have different parameter spaces, we can run into problems, since they will certainly have different prior distributions, even if both of them are uniform.

As a simple example, consider the case of fitting a polynomial. Imagine our probabilistic model is as follows. Given a set of points \( \{(x_1, y_1), (x_2, y_2), \ldots (x_k, y_k)\} \), where each \( x_i \) is an independent variable and the dependent probability \( p(y_i) \) is a normal distribution with variance 1 centered at \( f(x_i) \). Now, if we force \( f(\cdot) \) to be a linear polynomial, with only two parameters, we will arrive at some probability for the observed data, given the model. If we allow \( f(\cdot) \) to be a quadratic, we will certainly get a higher probability, since the space of quadratic equations is strictly larger than the space of linear equations. If we were to correctly account for the prior probabilities of the models, we could perhaps correct for this, finding linear equations when appropriate.

Unfortunately, without some outside knowledge, we have only approximations for the prior probabilities of various models. In the presence of large datasets, however, the need to correctly account for the prior probability is somewhat mitigated. The probability of the observed events given the model grows ever smaller as the number of events increases, and hence plays an ever increasing role in the determination of the maximum likelihood model, compared to the prior, which does not change with more events. With sequences of millions of events, we hope that the product discussed above, \( \Pi_i p_i(e_i) \), will distinguish models sufficiently so that even given the most pessimistic assumptions about prior probabilities, the model with the larger product would win. For instance, if we imagine a sequence
of one million events, where the probability of each was 0.500 for one model and 0.501 for another, by the end of the sequence the difference in products would be a factor of $1E867$, likely dwarfing the difference in prior probabilities between the two models.

Once we accept this, the next task becomes searching for the maximum likelihood model under these conditions. In some cases, such as an exponential distribution for a single variable, we can find the maximum likelihood model analytically. In other cases, such as a power-law distribution with exponential cutoff, a search of the parameter space must be done numerically. An important computational note when doing this is that computing the products of millions of small probabilities will quickly lead to machine underflow. Luckily, there is a simple solution to this problem. Since $\log a \cdot b = \log a + \log b$, maximizing the product is the same as maximizing the sum of the logarithms, a task which will not normally lead to the same numerical problems.

One application for this method is in the fitting of power-law distributions. In this application, we have a sample of scalar values, and the model is that the likelihood of a value $x$ is proportional to $x^{-\alpha}$ for some $\alpha$. These distributions seem to appear everywhere, and when plotted on log-log scales, give roughly straight lines. If we look back at the data we generated via preferential attachment, the correct distribution is $\frac{4}{x(x+1)(x+2)}$, which is proportional to $x^{-3}$ for large $x$. However, samples with large $x$ are infrequent and play a small role in fitting the overall likelihood. Thus, if we have to fit to a single power-law distribution, we will claim a power less than 3. In particular, the value 2 occurs about a quarter as often as the value 1, suggesting a power of 2 not 3 and it turns out that when we actually fit this data, we arrive at 2.30. If we use other methods to fit the
Figure 2.2: Fitting a rough power-law distribution. The true distribution is $\frac{4}{x(x+1)(x+2)}$.

distribution, we may end up skewing things even more towards the lower values. Figure 2.2 shows the data along with the most likely (discrete) power-law fit and the best fit using standard least-squares fitting, which gives a power of 2.08. A final note is that we can try fitting using standard least-squares methods after log-transforming the data. While this is rather unprincipled, it gives us lines that are visually appealing on log-log plots, as long as we constrain the range of $x$ to be only those points for which we have sufficient data.
CHAPTER 3

GROUP FORMATION IN LARGE SOCIAL NETWORKS

The tendency of people to come together and form groups is inherent in the structure of society; and the ways in which such groups take shape and evolve over time is a theme that runs through large parts of social science research [32]. The study of groups and communities is also fundamental in the mining and analysis of phenomena based on sociological data — for example, the evolution of informal close-knit groups within a large organization can provide insight into the organization’s global decision-making behavior; the dynamics of certain subpopulations susceptible to a disease can be crucial in tracking the early stages of an epidemic; and the discussions within an Internet-based forum can be used to follow the emergence and popularity of new ideas and technologies. The digital domain has seen a significant growth in the scale and richness of on-line communities and social media, through the rise of social networking sites beginning with Friendster and its relatives, and continuing to more recent systems including MySpace, Facebook, and LiveJournal, as well as media-sharing sites such as Flickr.

Understanding the structure and dynamics of social groups is a natural goal for network analysis, since such groups tend to be embedded within larger social network structures. That is, given a collection of individuals linked in an underlying social network, the groups and communities that they identify with can be thought of as corresponding to subgraphs of this network, growing and overlapping one another in a potentially complex fashion. A group that grows mainly through the aggressive recruitment of friends by other friends would appear as a subgraph branching out rapidly over time along links in the network; a group in which the decision to join depends relatively little on the influence of friends might appear
instead as a collection of small disconnected components that grows in a “speckled” fashion.  

While abstract descriptions such as this — of groups growing concurrently and organically in a large network — are clearly suggestive, the fact is that it has been very hard to make concrete empirical statements about these types of processes. Much of the challenge arises from the difficulty in identifying and working with appropriate datasets: one needs a large, realistic social network containing a significant collection of explicitly identified groups, and with sufficient time-resolution that one can track their growth and evolution at the level of individual nodes. A further challenge has been the lack of a reasonable vocabulary for talking about group evolution — with each group growing in its own particular part of the network, how do we abstract and quantify the common types of patterns that we observe?

3.1 Analyzing Group Formation and Evolution

We seek to address these challenges, exploring the principles by which groups develop and evolve in large-scale social networks. We consider a number of broad principles about the formation of social groups, concerning the ways in which they grow and evolve, and we formalize concrete questions around them that can be tested on network data.

To do this, we take advantage of rich datasets and computational models for describing the process of group formation. In particular, as our primary sources

---

1While such social networks are not themselves directly observable, on-line systems can provide rich data on large networks of interactions that are highly reflective of these underlying social networks. As has become customary in the computer science community, we also refer to these observable networks as social networks, while recognizing that they are only a reflection of the complete picture of social interactions.
of data, we make use of two networks that combine the desirable features outlined above: LiveJournal, a social networking and blogging site with several million members and a large collection of explicit user-defined communities; and DBLP, a publication database with several hundred thousand authors over several decades, and where conferences serve as proxies for communities. We will say more about these datasets below; for now, we note the crucial point that we are focusing on networks where the members have explicitly identified themselves as belonging to particular groups or communities — we are thus not seeking to solve the unsupervised graph clustering problem of inferring “community structures” in a network (e.g., [56, 55, 59, 66, 101]), since for us the relevant communities have been identified by the members themselves.

We consider three main types of questions.

- **Membership.** What are the structural features that influence whether a given individual will join a particular group?

- **Growth.** What are the structural features that influence whether a given group will grow significantly (i.e. gain a large net number of new members) over time?

- **Change.** A given group generally exists for one or more purposes at any point in time; in our datasets, for example, groups are focused on particular “topics of interest.” How do such foci change over time, and how are these changes correlated with changes in the underlying set of group members?

The question of membership is closely related to the well-studied topic of diffusion of innovation in the social sciences (see e.g. [109, 115, 123] as well as [43, 70, 108] for more recent applications in the data mining literature). That is, if
we view the act of joining a particular group as a kind of behavior that “spreads” through the network, then how does one’s probability $p$ of joining a group depend on the friends that one already has in the group? Perhaps the most basic such question is how the probability $p$ depends on the number of friends $k$ that one already has in the group. This is a fundamental question in research on diffusion in social networks, and most mathematical models of this process implicitly posit a model for the dependence of $p$ on $k$ (see e.g. [43, 70, 123]); however, it has to date been easier to explore such models theoretically than to obtain reasonable estimates for them empirically on large-scale data. Here we find that this dependence is remarkably similar for groups in the LiveJournal and DBLP datasets, despite the very different meaning of the groups in these two domains; the probability $p$ increases, but sublinearly so, in the number of friends $k$ belonging to the group. The data suggest a “law of diminishing returns” at work, where having additional friends in a group has successively smaller effect but nonetheless continues to increase the chance of joining over a fixed time window. In the context of diffusion models this result is somewhat surprising, in that it does not appear to be explained well by models that posit logistic or “critical mass” behavior for $p$ versus $k$.

Beyond this, however, the available data makes possible a much broader investigation of membership in groups. While theoretical models of diffusion have focused primarily on just the effect of $k$, the number of friends one already has in a group, we would like to understand more generally the structural properties that are most influential in determining membership. Here we do this by applying a decision-tree approach to the question, incorporating a wide range of structural features characterizing the individual’s position in the network and the subgraph defining the group, as well as group features such as level of activity among members. In the process we find that the probability of joining a group depends in
subtle but intuitively natural ways not just on the number of friends one has, but also on the ways in which they are connected to one another.

To take one illustrative example: for moderate values of $k$, an individual with $k$ friends in a group is significantly more likely to join if these $k$ friends are themselves mutual friends than if they aren’t. This example fits naturally with known sociological dichotomies on diffusion, and hence it hints at some of the more qualitative processes at work in the communities we are studying.

We adopt a similar approach to the question of growth: given a group, how well can we estimate whether it will grow by a significant fraction of its current size over a fixed time period? We find that reasonable estimation performance can be obtained based purely on the structural properties of the group as a subgraph in the underlying social network. As with membership, relatively subtle structural features are crucial in distinguishing between groups likely to grow rapidly and those not likely to. Again, to focus on one example, groups with a very large number of triangles (consisting of three mutual friends) grow significantly less quickly overall than groups with relatively few triangles. Overall, then, the framework based on decision trees can be viewed as a way to identify the most “informative” structural and group features influencing the growth and membership processes, with the payoff that the resulting features have natural interpretations in terms of the underlying sociological considerations.

Groups not only grow and attract new members — the very characteristics of a group can change over time. A group $A$ may change its focus of interest to become more like some other group $B$; it may also change its membership to become more like $B$. The final set of questions that we investigate addresses issues of change in group membership and interests, as well as the extent to which there
is a correlation between these two types of change. For instance do changes in membership consistently precede or lag changes in interest? While such questions are extremely natural at a qualitative level, it is highly challenging to turn them into precise quantitative ones, even on data as detailed as we have here. We approach this through a novel methodology based on burst analysis [71]; we identify bursts both in term usage within a group and in its membership. We find that these are aligned in time to a statistically significant extent; furthermore, for CS conference data in DBLP, we present evidence that topics of interest tend to cross between conferences earlier than people do.
Figure 3.2: The probability $p$ of joining a DBLP community as a function of the number of friends $k$ already in the community. Error bars represent two standard errors.

### 3.2 Community Membership

Before turning to our studies of the processes by which individuals join communities in a social network, we provide some details on the two sources of data, LiveJournal and DBLP. LiveJournal (LJ) is a free on-line community with almost 10 million members; a significant fraction of these members are highly active. (For example, roughly 300,000 update their content in any given 24-hour period.) LiveJournal allows members to maintain journals, individual and group blogs, and — most importantly for our study here — it allows people to declare which other members are their friends and to which communities they belong. By joining a community, one typically gains the right to create new posts in that community and other people’s posts become more accessible.
DBLP, our second dataset, is an on-line database of computer science publications, providing the title, author list, and conference of publication for over 400,000 papers. A great deal of work has gone into disambiguation of similar names, so co-authorship relationships are relatively free of name resolution problems. For our purposes, we view DBLP as parallel to the friends-and-communities structure of LiveJournal, with a “friendship” network defined by linking people together who have co-authored a paper, and with conferences serving as communities. We say that a person has joined a community (conference) when he or she first publishes a paper there; and, for this section, we consider the person to belong to the community from this point onward. (See Section 3.4 for an analysis of changes in community membership that include notions of both joining and leaving.) For simplicity of terminology, we refer to two people in either of LJ or DBLP as “friends” when they are neighbors in the respective networks.

A fundamental question about the evolution of communities is determining who will join a community in the future. As discussed above, if we view membership in a community as a kind of “behavior” that spreads through the network, then we can gain initial insight into this question from the study of the diffusion of innovation [109, 115, 123].

### 3.2.1 Dependence on number of friends

An underlying premise in diffusion studies is that an individual’s probability of adopting a new behavior increases with the number of friends already engaging in the behavior — in this case, the number of friends already in the community.

In Figures 3.1 and 3.2 we show this basic relationship for LJ and DBLP respec-
Table 3.1: Features.

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Featuers related to the community, $C$. (Edges between only members of the community are $E_C \subseteq E$.)</td>
<td>Number of members ($</td>
</tr>
<tr>
<td>Features related to an individual $u$ and her set $S$ of friends in community $C$.</td>
<td>Number of friends in community ($</td>
</tr>
</tbody>
</table>

tively: the proportion $P(k)$ of people who join a community as a function of the number $k$ of their friends who are already members. For LJ, this is computed as follows.

- First, we took two snapshots of community membership, roughly one month apart.
- Then we find all triples $(u,C,k)$ such that
  - $C$ is a community, and
– u is a user who, at the time of the first snapshot, did not belong to C, and
– u had k friends in C at that time.

• \( P(k) \) is then the fraction of such triples \((u, C, k)\) for a given \( k \) such that \( u \) belonged to \( C \) at the time of the second snapshot.

The procedure for DBLP is analogous, except that we use a snapshot for each year, and determine the fraction of individuals who “join” a conference from one year to the next.

The plots for LJ and DBLP exhibit qualitatively similar shapes, dominated by a “diminishing returns” property in which the curve continues increasing, but more and more slowly, even for relatively large numbers of friends \( k \). This forms an interesting contrast to the “S-shaped” curve at the heart of many theoretical models of diffusion, in which the probability of adopting a new behavior follows a logistic function, with slow growth in adoption probability for small numbers of friends \( k \), rapid growth for moderate values of \( k \), and a rapid flattening of the curve beyond this point.

In fact, the curves do exhibit some slight but noticeable “S-shaped” behavior: While the plots mainly show sublinear increase, we observe that they each display a deviation for \( k = 0, 1, 2 \) — namely, \( P(2) > 2P(1) \) for both LJ and DBLP. In other words, the marginal benefit of having a second friend in a community is particularly strong. However the remainder of each plot exhibits diminishing returns as \( k \) increases; thus the deviation at \( k = 0, 1, 2 \) can be seen as a slight “S-shaped” effect before the sublinear behavior takes over. Focusing on the function \( P(k) \) for LJ, since the error bars are smaller here, we see that the curve continues
increasing even for quite large values of $k$. Indeed, there is a close fit to a function of the form $P(k) = a \log k + b$ for appropriate $a$ and $b$.

A key reason that the curve for LJ is quite smooth is that the amount of data used to generate it is very large: there are roughly half a billion pairs $(u, C)$ for which $u$, at the time of the first snapshot, did not belong to $C$ but had at least one friend in $C$. The analogous quantity for DBLP is 7.8 million, and the curve becomes noisy at much smaller values of $k$. This suggests that for computing $P(k)$ as a function of $k$ in the context of diffusion studies, a very large sample may be required to begin seeing the shape of the curve clearly.

We find it striking that the curves for LJ and DBLP have such similar shapes (including the deviations for $k = 0, 1, 2$), given that the types of communities represented by these two datasets have such different characteristics: joining a community is a relatively lightweight operation in LJ, requiring very little investment of effort, whereas the analogous act of joining in the case of the DBLP dataset requires authorship and acceptance of a conference paper.

Curves with a diminishing returns property were also recently observed in independent work of Leskovec et al. [85], in yet another different context — recommendation data for on-line purchases — although the curves in their case become noisier at smaller values of $k$. The probability of friendship as a function of shared acquaintances and shared classes also exhibits diminishing returns in the work of Kossinets and Watts [75]. It is an interesting question to look for common principles underlying the similar shapes of the curves in these disparate domains.
3.2.2 A broader range of features

While these curves represent a good start towards membership prediction, they estimate the probability of joining a community based on just a single feature — the number of friends an individual has in the community. We now consider a range of other features related both to the communities themselves and to the topology of the underlying network which could also, in principle, influence the probability of joining a community. By applying decision-tree techniques to these features we find that we can make significant advances in estimating the probability of an individual joining a community. Table 3.1 summarizes the features that we use. In addition to features related exclusively to the social network structure, we also generate simple features that serve as indicators of the activity level of a community in LJ (for example, the number of messages posted by members of the community).\(^2\) A recurring principle in our experimental set-up is the following: since our goal is to understand which features from a particular set of structural and activity-based features are most informative, we intentionally control the set of features available to our algorithms. For the strict goal of obtaining high prediction performance, there are other features that could be included that would be less informative for our current purposes.

We now discuss the exact structure of the sets over which we make predictions for both LJ and DBLP.

**LiveJournal** For the more detailed studies of membership prediction, we focused on a subset of 875 LJ communities, comparing them from the first LJ snapshot to

\(^2\)Due to the much more regimented nature of conference activity, we do not generate analogous activity features for the DBLP dataset.
the second.\textsuperscript{3} For the first of these snapshots, we also built the network structure on the communities and their fringes. (We define the \textit{fringe} of a community $C$ to be the set of all non-members of $C$ who have at least one friend in $C$.) In addition, we collected all posts during the two weeks prior to the initial snapshot. (This two-week period was disjoint from the initial period during which we selected the 875 communities.)

From this information, we created a data point $(u, C)$ for each user $u$ and community $C$ such that $u$ belonged to the fringe of $C$ in the first snapshot. We then estimated the probability each such fringe member would be in the community in the second snapshot. Note that this task is an instance of the general problem of estimating missing values in a matrix: we are given a matrix whose rows correspond to users, whose columns correspond to communities, and whose entries $(u, C)$ indicate whether $u$ joins $C$ in the time interval between the two snapshots. In this way, the set-up is syntactically analogous to what one sees for example in collaborative-filtering-style problems; there too one is attempting to estimate hidden matrix-valued data (e.g. which customers are likely to buy which books). In keeping with our design principle, however, we are interested in performance based only on carefully selected features of the users $u$ and communities $C$, rather than their actual identities.

We have 17,076,344 data points $(u, C)$, and of these, only 14,488 of represent instances in which user $u$ actually joined community $C$, for an average rate of $8.48e^{-4}$. Note that our task here, to estimate probabilities for individuals joining,

\textsuperscript{3}We chose the 875 communities as follows. We monitored all new posts to all communities during a 10 day period. Of those communities which had at least 1 post, we selected the 700 most active communities along with 300 at random from the others with at least 1 post. For technical reasons, it turned out that we were not able to collect accurate data on the largest of the communities, and hence were forced to discard communities which started with over 1000 members, leaving 875 communities.
is compatible with the low aggregate rate of joining. To make estimates about joining, we grow 20 decision trees. Each of the 875 communities is selected to have all of its fringe members included in the decision tree training set or not with independent probability 0.5. At each node in the decision tree, we examine every possible feature, and every binary split threshold for that feature. Of all such pairs, we select and install the split which produces the largest decrease in entropy [107] (i.e. information gain). We continue to install new splits until there are fewer than 100 positive cases at a node, in which case we install a leaf which predicts the ratio of positives to total cases for that node. Finally, for every case we find the set of decision trees for which that case was not included in the training set used to grow the tree. The average of these predictions gives us a prediction for the case. For the few cases that we include in the training set of every decision tree, we simply predict the baseline 8.48e-4. This technique of model averaging [23] has been shown to be effective in prediction settings such as these.

**DBLP** For DBLP we perform a very similar experiment. Here we define the fringe of a conference $C$ in year $y$ to be those people who have not published in $C$ prior to year $y$, but who have coauthored with at least one person who has published in $C$ prior to $y$. For every conference, year, and fringe member in that year we create a data point. Of 7,651,013 data points, we find that 71,618 correspond to individuals who join the conference (publish a paper in it) in the year in question. Again, to make predictions we use 20 simple decision trees grown in an identical way to those for LJ.
Table 3.2: Prediction performance for single individuals joining communities in LiveJournal. For every individual in the fringe of one of our 875 communities, we estimate the probability that person will join in a one-month interval. We repeat this experiment using 3 sets of features: only the number of friends in the community, features based on post activity (plus basic features: number of friends and community size), and finally the combination of all the features, including the graph-theoretic ones from Table 3.1.

<table>
<thead>
<tr>
<th>Features Used</th>
<th>ROCA</th>
<th>APR</th>
<th>CXE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Friends</td>
<td>0.69244</td>
<td>0.00301</td>
<td>0.00934</td>
</tr>
<tr>
<td>Post Activity</td>
<td>0.73421</td>
<td>0.00316</td>
<td>0.00934</td>
</tr>
<tr>
<td>All</td>
<td>0.75642</td>
<td>0.00380</td>
<td>0.00923</td>
</tr>
</tbody>
</table>

Table 3.3: Prediction performance for single individuals joining communities in DBLP. For every triple of a year, a conference, and an author who had not published in the conference, but had coauthored with a conference member, we estimate the probability that the author will publish in the conference’s next meeting.

<table>
<thead>
<tr>
<th>Features Used</th>
<th>ROCA</th>
<th>APR</th>
<th>CXE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Friends</td>
<td>0.64560</td>
<td>0.01236</td>
<td>0.06123</td>
</tr>
<tr>
<td>All</td>
<td>0.74114</td>
<td>0.02562</td>
<td>0.05808</td>
</tr>
</tbody>
</table>

### 3.2.3 Results and Discussion

Table 3.2 and Table 3.3 summarize the performance we achieve with these decision trees. For comparison, both tables contain the baseline performance one could achieve by predicting based solely on the number of friends a fringe member already has in the community. In all of our predictions, even the people who are most likely to join a community still have a probability much less than 50%. This makes performance metrics like accuracy meaningless, since if one had to make binary decisions, one would simply predict that no one would join. We thus use performance metrics that are based on the order of predictions: area under the ROC curve (ROCA) and average precision (APR), as well as cross entropy (CxEE), which treats predictions as probabilities. The two tables show that we are able to do significantly better by using features beyond the number of friends an individual has in the community.
Figure 3.3: The top two levels of decision tree splits for predicting single individuals joining communities in LiveJournal. The overall rate of joining is $8.48e^{-4}$.

**Internal Connectedness of Friends**  The top-level splits in the LJ and DBLP decision trees were quite stable over multiple samples; in Figure 3.3 we show the top two levels of splits in a representative decision tree for LJ. We now discuss a class of features that proved particularly informative for the LJ dataset: the internal connectedness of an individual’s friends.

The general issue underlying this class of feature is the following: given someone with $k$ friends in a community, are they more likely to join the community if many of their friends are directly linked to one another, or if very few of their friends are linked to one another? This distinction turns out to result in a significant effect on the probability of joining. To make this precise, we use the following notation. For an individual $u$ in the fringe of a community, with a set $S$ of friends in the community, let $e(S)$ denote the number of edges with both ends in $S$. (This is the number of pairs in $S$ who are themselves friends with each other.) Let $\varphi(S) = \frac{e(S)}{{|S| \choose 2}}$ denote the fraction of pairs in $S$ connected by an edge.

We find that individuals whose friends in a community are linked to one another — i.e., those for which $e(S)$ and $\varphi(S)$ are larger — are significantly more likely
to join the community. In particular, the top-level decision tree split for the LJ dataset is based on $\varphi(S)$, and in the right branch (when $\varphi(S)$ exceeds a lower bound), the next split is based on $e(S)$. We can see the effect clearly by fixing a number of friends $k$, and then plotting the joining probability as a function of $\varphi(S)$, over the sub-population of instances where the individual has $k$ friends in the community. Figure 3.4 shows this relationship for the sub-populations with $k = 3, 4, \text{ and } 5$; in each case, we see that the joining probability increases as the density of linkage increases among the individual’s friends in the community.

It is interesting to consider such a finding from a theoretical perspective — why should the fact that your friends in a community know each other make you more likely to join? As discussed in Section 2.4, there are sociological principles
that could potentially support either side of this dichotomy. On the one hand, arguments based on *weak ties* [62] (and see also the notion of *structural holes* in [26]) support the notion that there is an informational advantage to having friends in a community who do not know each other — this provides multiple “independent” ways of potentially deciding to join. On the other hand, arguments based on social capital (e.g. [33, 32]) suggest that there is a trust advantage to having friends in a community who know each other — this indicates that the individual will be supported by a richer local social structure if he or she joins. Thus, one possible conclusion from the trends in Figure 3.4 is that trust advantages provide a stronger effect than informational advantages in the case of LiveJournal community membership.

The fact that edges among one’s friends make community membership more likely is also consistent with observations made in recent work of Centola, Macy, and Eguiluz [29]. They contend that instances of successful social diffusion “typically unfold in highly clustered networks” [29]. In the case of LJ and DBLP communities, for example, Macy observes that links among one’s friends may contribute to a “coordination effect,” in which one receives a stronger net endorsement of a community if it is a shared focus of interest among a group of interconnected friends [91].

**Relation to Mathematical Models of Diffusion** There are a number of theoretical models for the diffusion of a new behavior in a social network, based on simple mechanisms in which the behavior spreads contagiously across edges; see for example [42, 70, 123] for references. Many of these models operate in regimented time steps: at each step, the nodes that have already adopted the behavior may

---

4We thank David Strang for helping to bring the arguments on each side into focus.
have a given probability of “infecting” their neighbors; or each node may have a given threshold $d$, and it will adopt the behavior once $d$ of its neighbors have adopted it.

Now, it is an interesting question to consider how these models are related to the empirical data in Figures 3.1 and 3.2. The theoretical models posit very simple dynamics by which influence is transmitted: in each time step, each node assesses the states of its neighbors in some fashion, and then takes action based on this information. The spread of a real behavior, of course, is more complicated, and our measurements of LJ and DBLP illustrate this: we observe the behavior of an individual node $u$’s friends in one snapshot, and then $u$’s own behavior in the next, but we do not know (i) when or whether $u$ became aware of these friends’ behavior, (ii) how long it took for this awareness to translate into a decision by $u$ to act, and (iii) how long it took $u$ to actually act after making this decision. (Imagine, for example, a scenario in which $u$ decides to join a community after seeing two friends join, but by the time $u$ actually joins, three more of her friends have joined as well.) Moreover, for any given individual in the LJ and DBLP data, we do not know how far along processes (i), (ii), and (iii) are at the time of the first snapshot — that is, we do not know how much of the information contained in the first snapshot was already known to the individual, how much they observed in the interval between the first and second snapshots, and how much they never observed.

These considerations help clarify what the curves in Figures 3.1 and 3.2 are telling us. The concrete property they capture is the measured probability of adoption over a fixed time window, based on observed properties of an earlier snapshot — and they do this for network data on a scale that has been hard to obtain in
earlier social science studies of this phenomenon. Building on this, it is a natural challenge to relate the data underlying these curves to more purely operational models by which influence is spread through a network, and potentially to assess whether such models are reasonable approximations of real diffusion processes.

3.3 Community Growth

We now turn to a different but related prediction task: identifying which communities will grow significantly over a given period of time. We apply decision tree techniques to this task as well, using the community features given in the first half of Table 3.1.

For this experiment, our features come from two snapshots of community membership and social network topology, taken roughly 4 months apart. Since the behavior of extremely small communities is determined by many factors that are not observable from the network structure, we perform our experiments only on those communities which had at least 100 members at the time of the first snapshot. We say that a community has a growth rate of $x\%$ if its size in the second snapshot is $x\%$ larger than its size in the first snapshot. Over all communities, the mean growth rate was 18.6%, while the median growth rate was 12.7%.

We cast this problem directly as a binary classification problem in which class 0 consists of communities which grew by less than 9%, while class 1 consists of communities which grew by more than 18%. We find that by excluding the middle we achieve more meaningful estimates of performance, as it is unreasonable to expect good performance in the region around a simple threshold. This leaves us a data set with 13570 communities, 49.4% of which are class 1.
To make predictions on this dataset we again use binary decision trees. Because this data set is smaller and more balanced, we install binary splits until a node has less than 50 data points, in which case we install a leaf which predicts the fraction of positive instance at that point. We grow 100 decision trees on 100 independent samples of the full dataset. For a particular test case, we make a prediction for that case using all of the decision trees which were not grown using that case.

### 3.3.1 Results

For comparison, we start by considering a number of simple baseline predictions, shown in Table 3.4. Using the same technique of averaging trees, but with only a single feature, we construct three baselines. The first feature for comparison is simply the size of the community. One might suspect that communities with a large number of people became large for a reason and are likely to continue growing. The second baseline uses the number of people in the fringe of the community, as these are the people most likely to join. Finally, we use the ratio of these two
Figure 3.6: The rate of community growth as a function of the ratio of closed to open triads: having a large density of closed triads (triangles) is negatively related to growth. Error bars represent two standard errors.

Table 3.4: Results for predicting community growth: baselines based on three different features, and performance using all features.

<table>
<thead>
<tr>
<th>Features Used</th>
<th>ROCA</th>
<th>APR</th>
<th>CXE</th>
<th>ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fringe</td>
<td>0.55874</td>
<td>0.53560</td>
<td>1.01565</td>
<td>0.54451</td>
</tr>
<tr>
<td>Community Size</td>
<td>0.52096</td>
<td>0.52009</td>
<td>1.01220</td>
<td>0.51179</td>
</tr>
<tr>
<td>Ratio of Fringe to Size</td>
<td>0.56192</td>
<td>0.56619</td>
<td>1.01113</td>
<td>0.54702</td>
</tr>
<tr>
<td>Combination of above 3</td>
<td>0.60133</td>
<td>0.60463</td>
<td>0.98303</td>
<td>0.57178</td>
</tr>
<tr>
<td>All Features</td>
<td>0.77070</td>
<td>0.77442</td>
<td>0.82008</td>
<td>0.70035</td>
</tr>
</tbody>
</table>

features — the size of the fringe divided by the size of the community — as well as a combination of all three.

Table 3.4 shows that none of these simple features gives good performance by itself. While they each perform slightly better than random guessing, the difference is small. Furthermore, using these three baseline features in combination still does not yield very impressive results: an ROCA of 0.60133 as compared to 0.5 for
random guessing.

By including the full set of features described previously, however, we find that we can make predictions with reasonably good performance. ROCA increases to 0.77070, while accuracy goes up to 70%. Other performance metrics indicate improvement on similar scales. Furthermore, accuracy on the fastest growing communities is as high as 80%.

3.3.2 Discussion of Results

It is informative to look at which features are being used at the top-level splits made by the decision trees. Figure 3.5 shows the top 2 splits installed by a representative tree. While the features and splits in the tree varied depending on the sample, the top 2 splits were quite stable, with only minor variations between samples. The first of these is the number of people that have a large number of friends in the community. Given the results of the previous section, this is intuitively natural. At the top level, we see that communities with a higher percentage of fringe members with at least 13 friends in the community are much more likely to be of class 1. Furthermore, of the communities with relatively few such fringe members, the next split is based on individuals with 7 friends in the community.

A second class of features, also important for community growth though for less intuitively apparent reasons, is the density of triangles. (See the right subtree in Figure 3.5.) Communities for which the ratio of closed to open triads is too high are unlikely to grow. Although this shows up strongly in the data (see also Figure 3.6), it is not entirely clear how to interpret this result. It is possible that a large density of triangles indicates a kind of “cliqueishness” that makes the
community less attractive to join; it is also possible that high triangle density is a sign of a community that stopped gaining new members at some point in the past and has subsequently been densifying, adding edges among its current set of members. We are currently pursuing further investigations to attempt to interpret the role of this feature more clearly.

3.4 Movement Between Communities

Having analyzed the membership and growth of communities, we now turn to the question of how people and topics move between communities. A fundamental question here is the degree to which people bring topics with them from one community to another, versus the degree to which topics arise in a community and subsequently attract people from other communities. In other words, given a set of overlapping communities, do topics tend to follow people, or do people tend to follow topics? We also investigate a related question: when people move into a community are they more or less likely than other members of the community to be participants in current and future “hot topics” of discussion in that community?

While these questions are intuitively very natural, it is a challenge to define sufficiently precise versions of them that we can make quantitative observations. Furthermore, any attempt to make these questions precise will involve certain simplifications and approximations, and we start by discussing the reasons behind some of our experimental design decisions. We use the DBLP data discussed in earlier sections, with conferences serving as the communities (limiting the data to 87 conferences for which there is DBLP data over at least a 15-year time period). Since DBLP includes paper titles, we take the words in titles as the raw data for
identifying topics in each community. There are a number of indications that the cumulative set of words in titles can serve, for our purposes here, as an effective proxy for top-level topics (see e.g. [71] and some of the discussion at the end of this section).

Informally, it is easy to think of individual instances where two conferences $B$ and $C$ seemed to move “closer together” over some period of years (for example, NIPS and ICML in the period 2000-2003 — an observation borne out by analysis of the data as well). We now define experiments that ask whether, in general over all such movement patterns, these movements are at the level of topics, people, or both — and if both, then which kind of movement tends to precede the other.

3.4.1 Time Series and Detected Bursts

Intuitively, it is possible for the same topic $x$ to be “hot” at each of two conferences $B$ and $C$ at the same time, even if $B$ and $C$ are not highly similar in any “global” sense. Many of the effects we are seeking to understand have more the former flavor (a shared hot topic) than the latter (global similarity), so we structure our definitions around this former notion.

**Term Bursts** For a given conference $C$ and a word $w$, we denote by $T_{w,C}(y)$ the fraction of paper titles at conference $C$ in year $y$ that contain the word $w$. $T_{w,C}$ can thus be viewed as the time series giving the frequency of word $w$ at $C$ over a sequence of years. For each time series $T_{w,C}$, we identify bursts in the usage of $w$ using a simple stochastic model for term generation that identifies intervals in which the usage can be approximated by a “burst rate” that is twice the average
rate [71]. This burst detection technique was used in [71] on the same DBLP title data, and was observed to be effective at identifying “hot topics” at conferences. The same technique has since been used for finding term bursts in a range of other domains, for instance in detecting current topics in blogs [79].

For our purposes, these burst intervals serve to identify the “hot topics” that indicate a focus of interest at a conference. We say that a word $w$ is hot at a given conference $C$ in a year $y$ if the year $y$ is contained in a burst interval of the time series $T_{w,C}$. (Note that being a hot term is a property of three things: a term, a conference, and a year.)

We also note an important caveat. Clearly it does not make sense to evaluate any single paper based on whether it happens to use a particular word in its title or not. All of our experimental findings based on burst analysis, however, only consider the frequencies of bursty words over large sets of papers, and will in all cases be supported by strong findings of statistical significance. In this way, the noise inherent in specific paper titles is being smoothed out by looking across large samples.

**Movement Bursts** Next, we need to define a corresponding notion for author movement, and some care is needed here. Unlike title terms, individual people appear quite sparsely at conferences; even someone who is a “member” of a given conference community will generally not publish there every year. Moreover, movement is asymmetric — there may be movement from a conference $B$ to a conference $C$ but not vice versa — and so we need to employ a notion that is different from a simple overlap measure.

First, we define someone to be a *member* of a conference in a given year $y$ if
they have published there in the 5 years leading up to $y$. (In contrast to previous sections, this definition allows someone to be a member of a conference and later not a member, which is crucial for the kinds of analysis we do here.) We then say that author $a$ moves into conference $C$ from conference $B$ in year $y$ when $a$ has a paper in conference $C$ in year $y$ and is a member of conference $B$ in year $y - 1$. Note that movement is a property of two conferences and a specific year, and further that although this measure of movement is asymmetric, it may sometimes hold in both directions.

Let $M_{B,C}(y)$ denote the fraction of authors at $C$ in year $y$ with the property that they are moving into $C$ from $B$. Thus, $M_{B,C}$ can be viewed as a time series representing author movement, and we use burst detection to find intervals of $y$ in which the value $M_{B,C}(y)$ exceeds the overall average by an absolute difference of .10.\(^5\) We refer to such an interval as a $B \rightarrow C$ movement burst.

We now have word burst intervals, identifying hot terms, and movement burst intervals, identifying conference pairs $B, C$ during which there was significant movement. We next discuss some experiments that investigate how these are aligned in time.

\(^5\)We use an additive difference instead of a multiplicative factor to generate the burst rate here: multiplicative burst rates tend to penalize time series with large averages, and we need these here since they correspond to conference pairs with a large baseline overlap that nonetheless experience a sharp increase. While nearby values give similar results, we use a difference of .10 to define the burst rate since it produces about 200 burst intervals that are of moderate length, about 4 years each, over all conference pairs $(B, C)$. By way of comparison, the word bursts average about 5 years in length.
Table 3.5: Fractions of papers containing hot terms. Papers contributing to a
movement burst contain elevated frequencies of currently and expired hot terms,
but lower frequencies of future hot terms.

<table>
<thead>
<tr>
<th></th>
<th>All Papers</th>
<th>Papers Contrib. to Movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. papers</td>
<td>99774</td>
<td>10799</td>
</tr>
<tr>
<td>Currently hot</td>
<td>0.3859</td>
<td>0.4391</td>
</tr>
<tr>
<td>Future hot</td>
<td>0.1740</td>
<td>0.1153</td>
</tr>
<tr>
<td>Expired hot</td>
<td>0.2637</td>
<td>0.3102</td>
</tr>
</tbody>
</table>

3.4.2 Papers Contributing to Movement Bursts

We first consider characteristics of papers associated with some movement burst
into a conference $C$; we find that they exhibit significantly different properties
from arbitrary papers at $C$. In particular, one crucial difference is in the extent to
which they use terms that are currently hot at $C$, and the extent to which they use
terms that will be hot at $C$ in the future. Given that movement bursts intuitively
represent increased participation from some other community, these differences will
provide a first perspective on the general question of whether topics are following
people, or whether people are following topics.

We make this precise as follows. First, we say that a paper appearing at a
conference $C$ in a year $y$ contributes to some movement burst at $C$ if one of its
authors is moving from some conference $B$ into $C$ in year $y$, and $y$ is part of a
$B \rightarrow C$ movement burst. These are precisely the papers that, intuitively, are part
of the elevated movement from other conferences into $C$. Now, it is natural to
ask whether these papers that contribute to movement bursts differ from arbitrary
papers in the way they use hot terms. Here we say that a paper uses a hot term if
one of the words in its title is hot for the conference and year in which it appears.

As a baseline, 38.59% of all papers use hot terms. (While this number is a
useful benchmark for relative comparisons, its actual magnitude can clearly be
affected by altering the settings of the burst detection parameters.) On the other hand, as shown in Table 3.5, 43.91% of all papers contributing to movement bursts use hot terms. This difference is statistically significant: if we consider a binary variable that is true .3859 of the time, then the probability of seeing a sample of size 10799 (the number of papers contributing to movement bursts) where the variable is true .4391 of the time is seen to be $< 10^{-15}$ using a Chernoff-Hoeffding bound.

Thus it is apparent that papers written by people who are part of a burst of authors moving into a conference are more likely to be about topics that are “hot”, or experiencing a burst, than is the case for papers in general.

Given that papers contributing to a movement burst exhibit an elevated usage of hot terms, it is natural to also ask whether they also contain an unusually high proportion of terms that will be hot at some point in the future, or that were hot at some point in the past. Specifically, we say that a paper at a conference $C$ in year $y$ uses a future hot term if it contains a word that will experience a burst at $C$ starting in some year $> y$; we say that it uses an expired hot term if it contains a word that experienced a burst at $C$ ending in some year $< y$. As shown in Table 3.5, we find that papers contributing to movement bursts in fact use expired hot terms at a significantly higher rate than arbitrary papers at the same conference (31.02% vs. 26.37%), but use future hot terms at a significantly lower rate (11.53% vs. 17.40%). Again, these differences are statistically significant at comparable levels.

Taken together these results support the notion that a burst of authors moving into a conference $C$ from some other conference $B$ are drawn to topics that are currently hot at $C$; but there is also evidence that this burst of authors produces
papers that are comparably impoverished in their usage of terms that will be hot in the future. In other words, any notion that they are “germinating” terms that will soon become hot at conference $C$ is not borne out by the data; in fact, the opposite appears to be true.

We now turn to a second set of experiments that explores this temporal alignment of movement and term bursts in a different way, but leading to qualitatively similar conclusions.

### 3.4.3 Alignment between Different Conferences

We say that conferences $B$ and $C$ are *topically aligned* in a year $y$ if some word $w$ is hot at both $B$ and $C$ in year $y$. (We will also say that $B$ and $C$ are topically aligned via $w$.) Note that topical alignment, like movement, is a property of two conferences and a specific year. Also, two conferences can be topically aligned even if their overall collections of papers are quite different; they need only share a single common focus, in the form of a hot term.

It is natural to expect that two conferences are more likely to be topically aligned in a given year if there is also a movement burst going on between them. We first show that this is indeed the case, a basic result establishing that movements of terms and people are indeed correlated. Specifically, over all triples $(B, C, y)$ such that there is a $B \rightarrow C$ movement burst containing year $y$, we find that 56.34\% have the property that $B$ and $C$ are topically aligned in year $y$. As a baseline, only 16.10\% of all triples $(B, C, y)$ have the property that $B$ and $C$ are topically aligned in year $y$. Thus, the presence of a movement burst between two conferences enormously increases the chance that they share a hot term.
Figure 3.7: Four patterns of author movement and topical alignment: in each of (a)-(d), the labeled arrows represent term burst intervals for a shared hot term in conferences B and C, and the square wave represents a $B \rightarrow C$ movement burst. In the terminology from the text, (a) is shared interest, (b) is colonization, (c) is exploration, and (d) is shared membership.

Given this, we are now in a position to ask one of the questions posed informally at the outset: do movement bursts or term bursts tend to come first? Specifically, whenever there is a $B \rightarrow C$ movement burst, we look at all hot terms $w$ such that $B$ and $C$ are topically aligned via $w$ in some year $y$ inside the movement burst. There are now three events of interest:

(i) the start of the burst for $w$ at conference $B$;
(ii) the start of the burst for $w$ at conference $C$; and
(iii) the start of the $B \rightarrow C$ movement burst.

Let us consider how these might occur in order relative to one another, with interpretations of each; the various orders are depicted schematically in Figure 3.7. We then discuss how frequently these orders actually occur in the data.

- $w$ bursts at both $B$ and at $C$ (in some order) before the $B \rightarrow C$ movement burst begins. (See Figure 3.7(a).) We call this pattern *shared interest*, since the topical alignment of $B$ and $C$ happens before they come closer together in membership.
Table 3.6: Frequency of the four patterns relating movement and topical alignment. B+ (resp. B−) denotes that the burst of w at B follows (resp. precedes) the B → C movement burst; and analogously for C.

<table>
<thead>
<tr>
<th></th>
<th>C+</th>
<th>C−</th>
</tr>
</thead>
<tbody>
<tr>
<td>B+</td>
<td>194 (0.6025)</td>
<td>32 (0.0994)</td>
</tr>
<tr>
<td>B−</td>
<td>35 (0.1087)</td>
<td>61 (0.1894)</td>
</tr>
</tbody>
</table>

- w bursts at B, then the B → C movement burst begins, and then w bursts at C. (See Figure 3.7(b).) We call this pattern colonization, since one can imagine the movement from B to C as having a “colonizing” effect, carrying the term w from B (where it was already hot) to C (where it becomes hot).
- w bursts at C, then the B → C movement burst begins, and then w bursts at B. (See Figure 3.7(c).) We refer to this pattern as exploration, since one can imagine the hot topic at C attracting authors from B; subsequent to this “exploration” from B, the term becomes hot at B as well.
- The B → C movement burst begins, after which w bursts at B and at C (in some order). (See Figure 3.7(d).) We refer to this pattern as shared membership, since B and C come closer together in membership before the topical alignment happens via the common hot term w.

We now consider the relative frequencies of these four patterns. Over all cases in which there was a topical alignment of B and C concurrent with a B → C movement burst, we remove from the tabulation those in which two of the three relevant burst intervals (for the term at each conference, and for the movement) began in the same year. This leaves us with 322 instances in total, which are divided over the four categories as shown in Table 3.6. 194 of the instances correspond to the shared interest pattern: the term burst in each conference precedes the movement burst. In other words, of the four patterns, shared interest is 50%
Figure 3.8: Projections of a subset of the conferences in three different years (1980, 1987, 2003) into a two-dimensional LSI space.

more frequent than the other three patterns combined. The next most frequent is shared membership, with 61 instances, followed by colonization and exploration with 35 and 32 respectively.

As with the previous set of experiments, we find that the intuitively appealing notion of authors from a conference $B$ “transplanting” hot terms to a new conference $C$ is not in fact the dominant type of movement in the data. Rather, it is much more frequent for conferences $B$ and $C$ to have a shared burst term that is already underway before the increase in author movement takes place.
3.5 Discussion

We have considered the ways in which communities in social networks grow over time — both at the level of individuals and their decisions to join communities, and at a more global level, in which a community can evolve in both membership and content. Even with very rich data, it is challenging to formulate the basic questions here, and we view the elaboration of further questions to be an interesting direction for future work.

The availability of complex datasets on communities in social networks, and their evolution over time, leads naturally to a search for more refined theoretical models. It will be interesting to connect standard theoretical models of diffusion in social networks to the kinds of data on community membership that one can measure in on-line systems such as LiveJournal. One class of questions was suggested at the end of Section 3.2 — forming accurate models for the asynchronous processes by which nodes become aware of their neighbors’ behavior and subsequently act on it. Another goal is to understand how even very simple diffusion models may change if we parametrize influence not just by the number of neighbors who have adopted a behavior, but by the internal connectedness of these neighbors, following the findings in Section 3.2.

It would be interesting to relate some of the techniques developed here, particularly on movement between communities, to latent-space models for social networks as studied in Hoff et al. [64] and Sarkar and Moore [111]. Even without the network aspect, the movements in content exposed by very simple latent-space techniques are quite suggestive. For example, Figure 3.8 shows a representation of conferences from the DBLP dataset, encoded as term vectors and projected into a two-dimensional vector space $X$ defined by Latent Semantic Indexing (LSI)
In each year, the set of conferences projects differently into $X$, and their collective motion over successive years provides some illustration of their changing relationships to one another. Such representations can clearly form the basis for alternate ways of quantifying community movement, with conferences forming natural groupings by topic, and with certain parts of the space becoming “filled out” as particular areas emerge over time.

## 3.6 Related Work

As discussed above, there is a large body of work on identifying tightly-connected clusters within a given graph (see e.g. [56, 54, 59, 66, 101]). While such clusters are often referred to as “communities”, it is important to note that this is a very different type of problem from what we consider here — while this clustering work seeks to infer potential communities in a network based on density of linkage, we start with a network in which the communities of interest have already been explicitly identified and seek to model the mechanisms by which these communities grow and change. Dill et al. [39] study implicitly-defined “communities” of a different sort: For a variety of features (e.g. a particular keyword, a name of a locality, or a ZIP code), they consider the subgraph of the Web consisting of all pages containing this feature. Such communities of Web pages are still quite different from explicitly-identified groups where participants deliberately join, as we study here; moreover, the questions considered in [39] are quite different from our focus here.

The use of on-line social networking sites for data mining applications has been the subject of a number of recent papers; see [2, 89] for two recent examples. These
recent papers have focused on different questions, and have not directly exploited
the structure of the user-defined communities embedded in these systems. Studies
of the relationship between different newsgroups on Usenet [22, 124] has taken
advantage of the self-identified nature of these on-line communities, although again
the specific questions are quite different.

As noted earlier, the questions we consider are closely related to the diffusion
of innovations, a broad area of study in the social sciences [109, 115, 123]; the
particular property that is “diffusing” in our work is membership in a given group.
The question of how a social network evolves as its members’ attributes change
has been the subject of recent models by Sarkar and Moore [111] and Holme
and Newman [65]; a large-scale empirical analysis of social network evolution in a
university setting was recently performed by Kossinets and Watts [75]; and rich
models for the evolution of topics over time have recently been proposed by Wang
and McCallum [127]. Mathematical models for group evolution and change have
been proposed in a number of social science contexts; for an approach to this issue
in terms of diffusion models, we refer the reader to the book by Boorman and
Levitt [21].
In this chapter, we present a detailed study of network evolution by analyzing four large online social networks with full temporal information about node and edge arrivals. We study individual node arrival and edge creation processes that collectively lead to macroscopic properties of networks. Using a methodology based on the maximum-likelihood principle, we investigate a wide variety of network formation strategies, and show that edge locality plays a critical role in the evolution of networks. Our findings supplement earlier network models based on the inherently non-local preferential attachment.

Based on our observations, we develop a complete model of network evolution, where nodes arrive at a prespecified rate and select their lifetimes. Each node then independently initiates edges according to a "gap" process, selecting a destination for each edge according to a simple triangle-closing model free of any parameters. We show analytically that the combination of the gap distribution with the node lifetime leads to a power law out-degree distribution that accurately reflects the true network in all four cases. Finally, we give model parameter settings that allow automatic evolution and generation of realistic synthetic networks of arbitrary scale.

4.1 Models of Network Evolution

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

As discussed in Section 2.1, the observation of heavy-tailed degree distributions [52] led to hypothesis about edge creation processes (e.g., preferential attachment [8]) that could lead to this observation. In fact, there are several edge creation processes that all lead to heavy-tailed degree distributions and it is not clear which among them captures reality best.

Here we take a different approach. Instead of only focusing on the global network structure and then hypothesizing about what kind of microscopic node behavior would reproduce the observed macroscopic network structure, we focus directly on the microscopic node behavior per se. For the first time at such a large scale, we study a sequence of millions of individual edge arrivals, which allows us to directly evaluate and compare microscopic processes that give rise to global network structure.

4.1.1 Evaluation based on likelihood

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

Likelihood, discussed in Section 2.5, has not been considered to date in the
analysis of evolution of large social networks mainly due to lack of data and computational issues. Many early network datasets contained only a single or a small number of snapshots of the data, making likelihood computations for evolutionary models infeasible. We study four large social networks with exact temporal information about individual arrivals of millions of nodes and edges. Here we are therefore able to consider edge-by-edge evolution of networks, and hence efficiently compute the likelihood that a particular model would have produced a particular edge, given the current state of the network. In contrast to previous work on evolution of large networks that used a series of snapshots to consider patterns at global scale, we study the exact edge arrival sequence, which means we are able to directly observe and model the fine-grained network evolutionary processes that are directly responsible for global network patterns and statistics.

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

(1) Models may be compared directly in a unified way, rather than arguing whether faithful reproduction of, e.g., diameter is more important than clustering coefficient and so forth.

(2) As our understanding of real-world networks improves, the evaluation criterion, i.e., likelihood, remains unchanged while the generative models improve to incorporate the new understanding. Success in modeling can therefore be effectively tracked.

(3) Models may be meaningfully distinguished based on as-yet-undiscovered properties of real-world data.
4.1.2 Data and Model Structure

We consider four large online social network datasets — Flickr (flickr.com, a photo-sharing website), Delicious (del.icio.us, a collaborative bookmark tagging website), Yahoo! Answers (answers.yahoo.com, a knowledge sharing website), and LinkedIn (linkedin.com, a professional contacts website) — where nodes represent people and edges represent social relationships. These networks are large with up to millions of nodes and edges, and the time span of the data ranges from four months to almost four years. All the networks are in early stages of their evolution with the connected component being small and the clustering coefficient increasing over time.

We consider models that can be decomposed into three core processes, namely, the node arrival process (governs the arrival of new nodes into the network), the edge initiation process (determines for each node when it will initiate a new edge), and the edge destination selection process (determines the destination of a newly initiated edge). Our networks do not include removal of nodes or edges, so we do not model deletion (although we do model the “death” of a node in the sense that it ceases producing new edges).

4.1.3 Our results

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

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

Node arrival process. We find large variation in node arrival rates over the four networks, ranging from exponential to sub-linear growth. Thus we treat node arrival rate as input to our model.

Edge initiation process. Upon arrival, a node draws its lifetime and then keeps adding edges until reaching its lifetime, with edges inter-arrival rate following a power law with exponential cut-off distribution. We find that edge initiations are accelerating with node degree (age), and prove that this leads to power law out degree distributions. The model produces accurate fits and high likelihood.

Edge destination selection process. We find that most edges (30%–60%) are local as they close triangles, i.e., the destination is only two hops from the source. We consider a variety of triangle-closing mechanisms and show that a simple scheme, where a source node chooses an intermediate node uniformly from among its neighbors, and then the intermediate node does the same, has high likelihood.

Our model is simple and easy to implement. It precisely defines the network
Table 4.1: Network dataset statistics. $E_b$ is the number of bidirectional edges, $E_u$ is the number of edges in undirected network, $E_\Delta$ is the number of edges that close triangles, $\%$ is the fraction of triangle-closing edges, $\rho$ is the densification exponent ($E(t) \propto N(t)^\rho$), and $\kappa$ is the decay exponent ($E_h \propto \exp(-\kappa h)$) of the number of edges $E_h$ closing $h$ hop paths (see Section 4.4 and Figure 4.4).

<table>
<thead>
<tr>
<th>Network</th>
<th>$T$</th>
<th>$N$</th>
<th>$E$</th>
<th>$E_b$</th>
<th>$E_u$</th>
<th>$E_\Delta$</th>
<th>$%$</th>
<th>$\rho$</th>
<th>$\kappa$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flickr</td>
<td>621</td>
<td>584,207</td>
<td>3,554,130</td>
<td>2,594,078</td>
<td>2,257,211</td>
<td>1,475,345</td>
<td>65.63</td>
<td>1.32</td>
<td>1.44</td>
</tr>
<tr>
<td>Delicious</td>
<td>292</td>
<td>203,234</td>
<td>430,707</td>
<td>348,437</td>
<td>348,437</td>
<td>96,387</td>
<td>27.66</td>
<td>1.15</td>
<td>0.81</td>
</tr>
<tr>
<td>Answers</td>
<td>121</td>
<td>598,314</td>
<td>1,834,217</td>
<td>1,067,021</td>
<td>1,300,698</td>
<td>303,858</td>
<td>23.36</td>
<td>1.25</td>
<td>0.92</td>
</tr>
<tr>
<td>LinkedIn</td>
<td>1294</td>
<td>7,550,955</td>
<td>30,682,028</td>
<td>30,682,028</td>
<td>30,682,028</td>
<td>15,201,596</td>
<td>49.55</td>
<td>1.14</td>
<td>1.04</td>
</tr>
</tbody>
</table>

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

4.2 Preliminaries

Datasets. For each of our four large network datasets, we know the exact time of all the node/edge arrivals. Table 4.1 gives the basic statistics of the four networks. All the networks slowly densify with a densification exponent [87] $\rho \approx 1.2$. All the networks, except Delicious, have shrinking diameter. In Flickr, Answers, and LinkedIn, the effective diameter reaches the maximum value of 10 when the network has around 50,000 nodes, and then slowly decreases to the around 7.5; in Delicious, the diameter is practically constant. Also, in all the networks, a majority of edges are bidirectional (column $E_b$). The reciprocity is 73% in Flickr, 81% in Delicious, and 58% in Answers; LinkedIn is undirected, but we know the edge initiator. The fraction of nodes that belongs to the largest weakly connected component is 69% in Flickr, 72% in Delicious, 81% in Answers, and
91% in LinkedIn.

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

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

### 4.3 Preferential attachment

In this section we study the bias in selection of an edge’s source and destination based on the degree and age of the node.
Figure 4.1: Probability $p_e(d)$ of a new edge $e$ choosing a destination at a node of degree $d$.

### 4.3.1 Edge attachment by degree

The preferential attachment (PA) model [8], discussed in Section 2.1, postulates that when a new node joins the network, it creates a constant number of edges, where the destination node of each edge is chosen proportional to the destination’s degree. Using our data, we compute the probability $p_e(d)$ that a new edge chooses a destination node of degree $d$; $p_e(d)$ is normalized by the number of nodes of degree $d$ that exist just before this step. We compute:

$$p_e(d) = \frac{\sum \mathbb{1}[e_t = (u, v) \land d_{t-1}(v) = d]}{\sum \mathbb{1}[u : d_{t-1}(u) = d]}.$$

First, Figure 4.1(a) shows $p_e(d)$ for the Erdős–Rényi [49] random network, $G_{np}$, with $p = 12/n$. In $G_{np}$, since the destination node is chosen independently of its degree, the line is flat. Similarly, in the PA model, where nodes are chosen proportionally to their degree, we get a linear relationship $p_e(d) \propto d$; see Figure 4.1(b).
Figure 4.2: Average number of edges created by a node of age \( a \).

Next we turn to our four networks and fit the function \( p_e(d) \propto d^\tau \). In Flickr, Figure 4.1(c), degree 1 nodes have lower probability of being linked as in the PA model; the rest of the edges could be explained well by PA. In Delicious, Figure 4.1(d), the fit nicely follows PA. In Answers, Figure 4.1(e), the presence of PA is slightly weaker, with \( p_e(d) \propto d^{0.9} \). LinkedIn has a very different pattern: edges to the low degree nodes do not attach preferentially (the fit is \( d^{0.6} \)), whereas edges to higher degree nodes are more “sticky” (the fit is \( d^{1.2} \)). This suggests that high-degree nodes in LinkedIn get super-preferential treatment. To summarize, even though there are minor differences in the exponents \( \tau \) for each of the four networks, we can treat \( \tau \approx 1 \), meaning, the attachment is essentially linear.
4.3.2 Edges by the age of the node

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

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

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

where $t_\ell$ is the time when the last node in the network joined.

Notice a spike at nodes of age 0. These correspond to the people who receive an invite to join the network, create a first edge, and then never come back. For all other ages, the level of activity seems to be uniform over time, except for LinkedIn, in which activity of older nodes slowly increases over time.

4.3.3 Bias towards node age and degree

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

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

- **DR**: With probability $\tau$, the node $v$ is selected preferentially (proportionally
Figure 4.3: Log-likelihood of an edge selecting its source and destination node. Arrows denote $\tau$ at highest likelihood.

to its degree), and with probability $(1 - \tau)$, uniformly at random: $\tau \cdot d_t(v) + (1 - \tau) \cdot 1/N(t)$.

- **A:** The probability of selecting a node is proportional to its age raised to power $\tau$: $a_t(v)^\tau$

- **DA:** The probability of selecting a node $v$ is proportional the product of its current degree and its age raised to the power $\tau$: $d_t(v) \cdot a_t(v)^\tau$.

Figure 4.3 plots the log-likelihoods under different models, as a function of $\tau$. The red curve plots the log-likelihood of selecting a source node and the green curve for selecting the destination node of an edge.
In Flickr the selection of destination is purely preferential: model D achieves the maximum likelihood at $\tau = 1$, and model DA is very biased to model D, i.e., $\tau \approx 1$. Model A has worse likelihood but model DA improves the overall log-likelihood by around 10%. Edge attachment in Delicious seems to be the most “random”: model D has worse likelihood than model DR. Moreover the likelihood of model DR achieves maximum at $\tau = 0.5$ suggesting that about 50% of the Delicious edges attach randomly. Model A has better likelihood than the degree-based models, showing edges are highly biased towards young nodes. For Answers, models D, A, and DR have roughly equal likelihoods (at the optimal choice of $\tau$), while model DA further improves the log-likelihood by 20%, showing some age bias. In LinkedIn, age-biased models are worse than degree-biased models. We also note strong degree preferential bias of the edges. As in Flickr, model DA improves the log-likelihood by 10%.

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

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

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

For \(G_{np}\) most new edges span nodes that were originally six hops away, and
then the number decays polynomially in the hops. In the PA model, we see a lot of long-range edges; most of them span four hops but none spans more than seven.

The hop distributions corresponding to the four real-world networks look similar to one another, and strikingly different from both \( G_{np} \) and PA. The number of edges decays exponentially with the hop distance between the nodes (see Table 4.1 for fitted decay exponents \( \kappa \)). This means that most edges are created locally between nodes that are close. The exponential decay suggests that the creation of a large fraction of edges can be attributed to locality in the network structure, namely most of the times people who are close in the network (e.g., have a common friend) become friends themselves.

These results involve counting the number of edges that link nodes certain distance away. In a sense, this overcounts edges \((u, w)\) for which \(u\) and \(w\) are far away, as there are many more distant candidates to choose from — it appears that the number of long-range edges decays exponentially while the number of long-
range candidates grows exponentially. To explore this phenomenon, we count the number of hops each new edge spans but then normalize the count by the total number of nodes at \( h \) hops. More precisely, we compute

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

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

Figures 4.5(c)–(f) show the plots for the four networks. Notice the probability of linking to a node \( h \) hops away decays double-exponentially, i.e., \( p_e(h) \propto \exp(\exp(-h)) \), since the number of edges at \( h \) hops increases exponentially with \( h \). This behavior is drastically different from both the PA and \( G_{np} \) models. Also note that almost all of the probability mass is on edges that close length-two paths. This means that edges are most likely to close triangles, i.e., connect people with common friends.

Column \( E_\Delta \) in Table 4.1 further illustrates this point by presenting the number of triangle-closing edges. Flicker and LinkedIn have the highest fraction of triangle-closing edges, whereas Answers and Delicious have substantially less such edges. Note that here we are not measuring the fraction of nodes participating in triangles. Rather, we unroll the evolution of the network, and for every new edge check to see if it closes a new triangle or not.
4.4.1 Triangle-closing models

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

To improve upon this baseline model we consider various models of choosing node $w$. We consider processes in which $u$ first selects a neighbor $v$ according to some mechanism, and $v$ then selects a neighbor $w$ according to some (possibly different) mechanism. The edge $(u, w)$ is then created and the triangle $(u, v, w)$ is closed. The selection of both $v$ and $w$ involves picking a neighbor of a node. We consider five different models to pick a neighbor $v$ of $u$, namely, node $v$ is chosen

- **random**: uniformly at random,

- **deg$^\tau$**: proportional to degree raised to power $\tau$, $d(v)^\tau$,

- **com**: prop. to the number of common friends $c(u, v)$ with $u$,
- \textbf{last}^\tau$: proportional to the time passed since \( v \) last created an edge raised to power \( \tau \),

- \textbf{comlast}^\tau$: proportional to the product of the number of common friends with \( u \) and the last activity time, raised to power \( \tau \).

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

Table 4.2 shows the percent improvement of various triangle-closing models over the log-likelihood of choosing a two-hop neighbor uniformly at random as a destination of the edge (the baseline). The simplest model, \textbf{random-random}, works remarkably well and has many desirable properties. It gives higher probability to nodes with more length-two paths, discounting each path by roughly \( 1/d(v) \). Moreover, it is also biased towards high-degree nodes, as they have multiple paths leading towards them.

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

The \textbf{com} model considers the strength of a tie between \( u \) and \( v \), which we
Table 4.2: Triangle-closing models. First pick intermediate node $v$ (fix column), then target node $w$ (fix row). The cell gives percent improvement over the log-likelihood of picking a random node two hops away (baseline).

<table>
<thead>
<tr>
<th></th>
<th>random</th>
<th>$\text{deg}^{0.2}$</th>
<th>com</th>
<th>last$^{-0.4}$</th>
<th>comlast$^{-0.4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flickr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random</td>
<td>13.6</td>
<td>13.9</td>
<td>14.3</td>
<td>16.1</td>
<td>15.7</td>
</tr>
<tr>
<td>$\text{deg}^{0.1}$</td>
<td>13.5</td>
<td>14.2</td>
<td>13.7</td>
<td>16.0</td>
<td>15.6</td>
</tr>
<tr>
<td>last$^{0.2}$</td>
<td>14.7</td>
<td>15.6</td>
<td>15.0</td>
<td>17.2</td>
<td>16.9</td>
</tr>
<tr>
<td>com</td>
<td>11.2</td>
<td>11.6</td>
<td>11.9</td>
<td>13.9</td>
<td>13.4</td>
</tr>
<tr>
<td>comlast$^{0.1}$</td>
<td>11.0</td>
<td>11.4</td>
<td>11.7</td>
<td>13.6</td>
<td>13.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>random</th>
<th>$\text{deg}^{0.3}$</th>
<th>com</th>
<th>last$^{-0.2}$</th>
<th>comlast$^{-0.2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delicious</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random</td>
<td>11.7</td>
<td>12.4</td>
<td>13.8</td>
<td>13.2</td>
<td>15.1</td>
</tr>
<tr>
<td>$\text{deg}^{0.2}$</td>
<td>12.2</td>
<td>12.8</td>
<td>14.3</td>
<td>13.7</td>
<td>15.6</td>
</tr>
<tr>
<td>last$^{-0.3}$</td>
<td>13.8</td>
<td>14.6</td>
<td>16.0</td>
<td>15.3</td>
<td>17.2</td>
</tr>
<tr>
<td>com</td>
<td>13.6</td>
<td>14.4</td>
<td>15.8</td>
<td>15.2</td>
<td>17.1</td>
</tr>
<tr>
<td>comlast$^{-0.2}$</td>
<td>14.7</td>
<td>15.6</td>
<td>16.9</td>
<td>16.3</td>
<td>18.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>random</th>
<th>$\text{deg}^{0.3}$</th>
<th>com</th>
<th>last$^{-0.2}$</th>
<th>comlast$^{-0.2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random</td>
<td>6.80</td>
<td>10.1</td>
<td>11.8</td>
<td>9.70</td>
<td>13.3</td>
</tr>
<tr>
<td>$\text{deg}^{0.2}$</td>
<td>7.18</td>
<td>10.5</td>
<td>12.2</td>
<td>10.1</td>
<td>13.7</td>
</tr>
<tr>
<td>last$^{-0.3}$</td>
<td>9.95</td>
<td>13.4</td>
<td>15.0</td>
<td>12.8</td>
<td>16.4</td>
</tr>
<tr>
<td>com</td>
<td>6.82</td>
<td>10.3</td>
<td>11.8</td>
<td>9.80</td>
<td>13.4</td>
</tr>
<tr>
<td>comlast$^{0.2}$</td>
<td>7.93</td>
<td>11.5</td>
<td>12.9</td>
<td>10.9</td>
<td>14.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>random</th>
<th>$\text{deg}^{0.1}$</th>
<th>com</th>
<th>last$^{-0.1}$</th>
<th>comlast$^{-0.1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LinkedIn</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random</td>
<td>16.0</td>
<td>16.5</td>
<td>18.2</td>
<td>17.2</td>
<td>18.5</td>
</tr>
<tr>
<td>$\text{deg}^{0.1}$</td>
<td>15.9</td>
<td>16.4</td>
<td>18.0</td>
<td>17.0</td>
<td>18.4</td>
</tr>
<tr>
<td>last$^{-0.1}$</td>
<td>19.0</td>
<td>19.5</td>
<td>21.1</td>
<td>20.0</td>
<td>21.4</td>
</tr>
</tbody>
</table>

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

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

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

4.5 Node and edge arrival process

In this section we turn our focus to the edge initiation process that determines which node is responsible for creating a new edge (Section 4.5.1), and then to the process by which new nodes arrive into the network (Section 4.5.2).

4.5.1 Edge initiation

In the following we assume that the sequence and timing of node arrivals is given, and we model the process by which nodes initiate edges. We begin by studying how long a node remains active in the social network, and then during this active lifetime, we study the specific times at which the node initiates new edges.
Figure 4.7: Exponentially distributed node lifetimes.

Node lifetime

To avoid truncation effects, we only consider those nodes whose last-created edge is in the first half of all edges in the data. Recall that the lifetime of a node $u$ is $a(u) = t_{d(u)}(u) - t_1(u)$. We evaluate the likelihood of various distributions and observe that node lifetimes are best modeled by an exponential distribution, $p_\ell(a) = \lambda \exp(-\lambda a)$. Figure 4.7 gives the plot of the data and the exponential fits, where time is measured in days. In Table 4.5, the row corresponding to $\lambda$ gives the values of fitted exponents. We note that the exponential distribution does not fit well the nodes with very short lifetimes, i.e., nodes that are invited into the network, create an edge and never return. But the distribution provides a very clean fit for nodes whose lifetime is more than a week.
Table 4.3: Edge gap distribution: percent improvement of the log-likelihood at MLE over the exponential distribution.

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

**Time gap between the edges**

Now that we have a model for the lifetime of a node $u$, we must model that amount of elapsed time between edge initiations from $u$. Let $\delta_u(d) = t_{d+1}(u) - t_d(u)$ be the time it takes for the node $u$ with current degree $d$ to create its $(d+1)$-st out-edge; we call $\delta_u(d)$ the *edge gap*. Again, we examine several candidate distributions to model edge gaps. Table 4.3 shows the percent improvement of the log-likelihood at the MLE over the exponential distribution. The best likelihood is provided by a power law with exponential cutoff: $p_g(\delta(d); \alpha, \beta) \propto \delta(d)^{-\alpha} \exp(-\beta \delta(d))$, where $d$ is the current degree of the node. (Note that the distribution is neither exponential nor Poisson, as one might be tempted to assume.) We confirm these results in Figure 4.8, in which we plot the MLE estimates to gap distribution $\delta(1)$, i.e., distribution of times that it took a node of degree 1 to add the second edge. In fact, we find that all gaps distributions $\delta(d)$ are best modeled by a power law with exponential cut-off (Table 4.3 gives improvements in log-likelihoods for $d = 1, \ldots, 5$ and the average for $d = 1, \ldots, 20$.)

For each $\delta(d)$ we fit a separate distribution and Figure 4.9 shows the evolution of the parameters $\alpha$ and $\beta$ of the gap distribution, as a function of the degree.
Figure 4.8: Edge gap distribution for a node to obtain the second edge, $\delta(1)$, and MLE power law with exponential cutoff fits.

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

Given the above observation, a natural hypothesis would be that nodes that will attain high degree in the network are in some way a priori special, i.e., they
Figure 4.9: Evolution of the $\alpha$ and $\beta$ parameters with the current node degree $d$. 
$\alpha$ remains constant, and $\beta$ linearly increases.

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

4.5.2 Node arrivals

Finally, we turn to the question of modeling node arrivals into the system. Figure
4.10 shows the number of users in each of our networks over time, and Table 4.4
captures the best fits. FICKR grows exponentially over much of our network,
while the growth of other networks is much slower. DELICIOUS grows slightly su-
perlinearly, LINKEDIN quadratically, and ANSWERS sublinearly. Given these wild
variations we conclude the node arrival process needs to be specified in advance as it varies greatly across networks due to external factors.

### 4.6 A network evolution model

We first take stock of what we have measured and observed so far. In Section 4.5.2, we analyzed the node arrival rates and showed that they are network-dependent.
and can be succinctly represented by a node arrival function $N(t)$ that is either a polynomial or an exponential. In Section 4.5.1, we analyzed the node lifetimes and showed they are exponentially distributed with parameter $\lambda$. In Section 4.3.1, we argued that the destination of the first edge of a node is chosen proportional to its degree (i.e., preferentially attached). In Section 4.5.1, we analyzed the time gaps between edge creation at a node and showed they can be captured by a power law with exponential cutoff, with parameters $\alpha, \beta$. In Section 4.4, we showed that most of the edges span two hops, and the simple random-random triangle-closing model works well.

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

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

The values of $N(\cdot)$ for the four networks are given in Table 4.4 and the values of $\alpha, \beta, \lambda$ are given in Table 4.5.
Note that one could also use more sophisticated edge placement techniques, like random surfer model [16] or other triangle-closing techniques as discussed in Section 4.4.1. For example, in step 5, a node \( u \) can pick a sequence of nodes \((u = w_0, w_1, \ldots, w_k = w)\), where each \( w_i \) is picked uniformly from the neighbors of \( w_{i-1} \), and the sequence length \( k \) is chosen from the distribution in Figure 4.4. Node \( u \) then links to \( w \).

### 4.6.1 Gaps and power law degree distribution

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

**Theorem 4.6.1.** The out-degrees are distributed according to a power law with exponent \( 1 + \frac{\lambda \Gamma(2-\alpha)}{\beta \Gamma(1-\alpha)} \).

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

\[
Z = \int_0^\infty \delta^{-\alpha} e^{-\beta \delta} d\delta = \frac{\Gamma(1-\alpha)}{(\beta d)^{1-\alpha}}. \tag{4.1}
\]

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

\[
\sum_{d=1}^{D} \delta(k) \leq a. \tag{4.2}
\]
Analogous to (4.1), we obtain the expected time gap \( E(\delta|d;\alpha,\beta) \) for a node of degree \( d \):

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

(4.3)

Combining (4.2) and (4.3), we relate the lifetime \( a \) and the expected final degree \( D \) of a node:

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

(4.4)

Notice that \( \sum_{d=1}^{D} d^{-1} = \Theta(\ln D) \). From (4.4), the final degree \( D \) of a node with lifetime \( a \) is

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

Thus, \( D \) is an exponential function of the age \( a \), i.e., \( D = r(a) = \exp(\mu a) \), where \( \mu = \frac{\Gamma(1-\alpha)}{\Gamma(2-\alpha)}\beta \). Since node lifetimes are exponentially distributed with parameter \( \lambda \), we now compute the distribution of \( D \) as a function of \( \lambda \) and \( \mu \) as follows:

\[
D \sim p_{\mu}(r^{-1}(D)) \left| \frac{d}{dD} r^{-1}(D) \right| = \frac{\lambda}{\mu D} e^{-\left(\frac{\lambda}{\mu}\right) \log D} = \frac{\lambda D^{-(1+\lambda/\mu)}}{\mu}.
\]

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

\( \square \)

**Validation of the model.** We validate the accuracy of our modeling assumptions by empirically estimating the lifetime \( \lambda \), and gap distribution \( \alpha,\beta \) parameter values for each network. We then apply Theorem 4.6.1, which yields the power-law degree exponents produced by our model. Then we empirically measure the true power law degree exponents of the four networks and compare them to predictions of Theorem 4.6.1. Table 4.5 shows the results. Note the predicted degree exponents remarkably agree with the true exponents, validating our model. This is interesting as we specified the model of temporal node behavior (lifetime+gaps) that results in an accurate structural network property (power-law degree distribution).
Table 4.5: Predicted by Theorem 4.6.1 vs true degree exponents.

<table>
<thead>
<tr>
<th></th>
<th>Flickr</th>
<th>Delicious</th>
<th>Answers</th>
<th>LinkedIn</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>0.0092</td>
<td>0.0052</td>
<td>0.019</td>
<td>0.0018</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.84</td>
<td>0.92</td>
<td>0.85</td>
<td>0.78</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.0020</td>
<td>0.00032</td>
<td>0.0038</td>
<td>0.00036</td>
</tr>
<tr>
<td>true</td>
<td>1.73</td>
<td>2.38</td>
<td>1.90</td>
<td>2.11</td>
</tr>
<tr>
<td>predicted</td>
<td>1.74</td>
<td>2.30</td>
<td>1.75</td>
<td>2.08</td>
</tr>
</tbody>
</table>

### 4.6.2 Unfolding network evolution

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

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

Figure 4.11 shows results for Flickr: clustering coefficient, degree distribution, and pairwise distance histogram for the true data, and the two simulations. The random-random model matches the true network well and outperforms than the PA process. Similar results also hold for other networks; we omit these plots for
Figure 4.11: We take Flickr network at first half of its evolution. Then we simulate the evolution using our model and PA for the second half, and compare the obtained networks with the real Flickr network. Notice our model matches the macroscopic statistical properties of the true Flickr network very well, and in fact much better than PA.

4.7 Discussion

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

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

4.8 Related Work

Many studies on online social networks, world wide web, and biological networks focused on macroscopic properties of static networks such as degree distributions, diameter, clustering coefficient, communities, etc; work in this area includes [52, 116, 7, 102, 44, 24]. Similarly, macroscopic properties of network evolution, like densification and shrinking diameters, were examined [87, 53, 104, 82].

Given that the classical Erdős–Rényi model cannot capture the above network characteristics, a number of alternate network models have been proposed. The copying [81] and the preferential attachment [8] (Section 2.1) models belong to this category. The Forest Fire model [87] attempts to explain the densification and decreasing-diameter phenomena observed in real networks. See [18] for a topic survey.

Recently, researchers examined the finer aspects of edge creation by focusing on a small set of network snapshots. The role of common friends in community formation was analyzed in Chapter 3. Kleinberg and Liben-Nowell [90] studied the predictability of edges in social networks. The role of triangle closure in social networks was long known to sociologists. Simmel theorized that people with common friends are more likely to create friendships and Krackhardt and Handcock [76] applied this theory to explain the evolution of triangle closures (see Section 2.2.1).
A network model based on closed triangles was proposed by Shi et al. [113].

The maximum-likelihood principle (Section 2.5) has been typically used to estimate network model parameters [88, 128, 131] or for model selection [14], which often requires expensive computations of high dimensional integrals over all possible node arrival sequences. In contrast, we use the likelihood in a much more direct way to evaluate and compare different modeling choices at a microscopic level.
Networks are only one source of Web data. While they provide an important framework for understanding the relationships between individuals, other data sources give us insights into other aspects of their online behavior. One of the most widely used applications on the Web is the search engine, and the sheer volume of queries issued each day makes this a very rich dataset which can be explored in a number of different directions. In this chapter, we will look specifically at the local aspect of search, correlating queries with geography.

There has been growing interest in local aspects of Web search, associating geographic information with Web content [9, 28, 40, 92, 93, 94] and search engine queries [63, 125]. Such applications point to the outlines of a broad and largely open issue: understanding and quantifying the types of spatial variation that search queries can exhibit.

Many topics have a geographic focus of interest; sports teams, newspapers, schools, airlines, cell-phone companies, politicians, tourist attractions, cuisine, hobbies, weather events, and styles of music are just a few examples. This diversity of examples exposes a corresponding diversity in the way that spatial variation can be manifested: interest in a topic can be tightly concentrated at a particular location or spread diffusely over a broader region; it can have one geographic “center” or several; it can move over time. To characterize queries according to this continuum of possible geographic traits, we need a model and a source of data rich enough to be able to discover subtle distinctions in spatial properties.

Here we describe a framework for modeling the spatial variation in search...
queries, using data from search engine query logs, supplemented with geolocation techniques to assign accurate locations to a (large) subset of the IP addresses issuing the queries. In this way, we define the geographic focus of a topic by the locations of the people who search for it, rather than the locations of the servers hosting the Web content itself — in other words, according to the locus of user interest expressed by searches.

Our model is probabilistic, and discovers for each query a maximum-likelihood value for a center — the “hot spot” of interest — and a dispersion — the extent to which interest in the query is tightly concentrated around the query or more diffuse. Each of these two quantities has a natural meaning: the center provides a location, while the dispersion situates the query on a continuous spectrum ranging from local to national appeal. In this way, they function similarity to the concepts of power and spread considered by Ding et al. in the context of Web resources [40], but defined in a very different way here based on a probabilistic model over usage data.

Determining an accurate center and dispersion has potential applications in a number of contexts. It clearly has a role in focusing search-based marketing and advertising efforts by region, based on geographic distinctions among different queries. It can also be useful as a component of search engine rankings themselves, for refining or reordering query results based on geographic information. Finally, it can help in tools concerned with tracking news and current awareness, by distinguishing among different interest in news topics by locale.

Ultimately, then, the question is whether there is enough signal in raw query logs to produce values for the center and dispersion of a query with reasonable accuracy. To take a concrete example, will the query “Yankees” really localize
to New York City — and will a much less searched-for term like “Royals” really localize to Kansas City — based purely on the latitudes and longitudes of queries, and despite the highly uneven distribution of locations from which these queries are being made, as well as the potentially obfuscating additional meanings of each? And will the range of dispersions for a topic such as baseball indeed distinguish teams (such as the Yankees) with a national following from those whose following is mainly local?

5.1 Basic properties of the model

We find, in fact, that a natural generative model for query production based on geography can produce highly accurate centers and dispersions for a broad range of queries, even reflecting geographic distinctions that are subtle, short-range, and sometimes temporally varying. The model is based on a decomposition of the surface of the earth into small grid cells; we assume that for each grid cell $x$, there is a probability $p_x$ that a random search from this cell will be equal to the query under consideration. In the basic form of the model, we then posit that each $p_x$ should decrease with the distance of $x$ from a “hot-spot” cell $z$; the cell $z$ is then the center, and the rate of decrease of $p_x$ as a function of its distance from $z$ determines the dispersion. We develop an efficient algorithm to compute the maximum-likelihood values for the center and dispersion, capable of scaling to handle complete query logs.

We describe a range of tests indicating that our method is effective at accurately localizing queries when there is a natural “ground truth” value for the center: to mention just a few examples, the names of almost all professional baseball teams are localized to their home cities, the names of almost all U.S. Senators are
localized to their home states, and the names of national parks are localized to their physical locations. Indeed, we show through a comparative evaluation that for localization accuracy on these types of queries, our probabilistic model significantly outperforms simpler geometric techniques, as well as state-of-the-art commercial software for query localization. This evaluation is based on the computed center; we also show that the dispersion follows a natural pattern for these classes of queries, ranging from queries that have broad appeal to those that are tightly focused geographically.

With a scalable method for determining centers and dispersions for queries, it becomes possible to assess the spatial variation of large collections of queries. We find that among the most 1000 frequent queries in the full log, there is surprising geographic concentration in a number of them. And with a simpler heuristic version of our probabilistic model, we perform a much more massive study — analyzing the 100,000 most frequent queries, and tagging each location on the surface of the earth with the queries that have the most significant concentration in that location with respect to the model. The resulting map of significant queries reveals trends that range from the global — such as the kinds of on-line social-networking sites favored in different parts of the world — to the very local — with striking geographic specificity as names of community colleges, high schools, and local newspapers vary between locations as little as ten miles apart.

5.1.1 Further extensions to the model

We can extend the model to handle other forms of spatial variation as well. To begin with, a number of queries have the property that their geographic focus noticeably shifts over time — for example, seasonal effects over slow time scales,
and news events over more rapid time scales. We show how to extend the model to allow the center and dispersion to vary with time — incorporating an additional probabilistic component that favors relatively “smooth” temporal motion in these quantities. Tracking these quantities over time can then be done efficiently with a shortest-path computation in a graph derived from the spatial arrangement of the grid cells from which the queries are issued. Here too the results can be quite striking: for the query “Hurricane Dean,” for example, one can see the query center moving day-by-day in a way that tracks the storm center’s westward movement into Mexico — and with a dispersion that starts out very concentrated but expands widely as the hurricane approaches land and begins appearing in the national news.

We describe other extensions as well, including a method for modeling spatial variation with multiple centers, and a method for comparing the geographic concentration of multiple queries to determine the approximate “sphere of influence” of each.

5.2 Modeling Spatial Variation

5.2.1 Methodology

Our data consists of Yahoo! search query logs. For each query, these logs give us both the query string and an approximation of the latitude and longitude from which the query originated (based on IP address). In order to reduce the effect of factors like network address translation, which allows many users to appear to come from the same IP, we first remove all queries that originate from IP addresses with particularly high search frequencies. The removed IPs account for less than
1% of all the queries. Furthermore, we focused only on North American queries, discarding any further west than 135° W, or further east than 60° W, as well as those south of 15° N or north of 60° N.

To reduce the variability introduced by people’s usage habits, we further process the data so that no IP address is considered to issue more than a single query during the time window we consider. Thus, with respect to a particular query, we consider how many distinct IP addresses from each geographic location issued at least one query during a time window, and how many of those IP addresses issued the particular query under consideration at least once. (For ease of discussion, we will often refer to “users” issuing queries, although for the above-mentioned reasons, IP addresses are in fact our basic unit of analysis.)

5.2.2 Model

For a given query, we posit a simple generative model to explain the differential frequency with which the query appears across geographic regions. In this model, each query has a geographic center represented by a single point. This center corresponds to the point at which the query should occur most frequently, with frequency then falling off in distance from the center.

In addition to its central point, each query in this model has two other parameters associated with it: a constant, $C$, giving the frequency at the query’s center, and an exponent $\alpha$ determining how quickly the frequency falls off as one gets further away from the center. The model posits that when a random user at distance $d$ from the query’s center issues a query, it is equal to the query under consideration with probability $Cd^{-\alpha}$. For queries that are very local, such as the
name of a small city, we expect a large value of $\alpha$, indicating that people rarely
search for that query unless they are quite close to the query’s center. On the
other hand, a query with essentially no geographic aspects or regional bias might
have $\alpha$ very close to zero, indicating that the frequency is essentially uniform over
geography. The polynomial functional form is employed here based on initial ex-
ploratory analysis of the data, and for tractability of the model; it is also sufficient
for our purposes, as it is capable of ranging smoothly (as $\alpha$ varies) from a uniform
distribution of user interest to distributions that are sharply peaked.

5.2.3 Algorithm

With this model in mind, we can focus on a particular query $q$ and take a maximum-
likelihood approach to discovering the parameters for $q$ from the data. For a
particular $C$, and $\alpha$, we can compute the probability that the true query logs
came from this model. For each log entry consisting of a user issuing a query, we
compute $p = C d^{-\alpha}$, where $d$ is that person’s distance from the query center. If
that person issues query $q$, we multiply the overall probability of the data by $p$,
and by $1 - p$ otherwise. (To avoid underflow, we work by adding logarithms of
probabilities, rather than actually multiplying.)

We now have a way to evaluate a particular set of parameters on a particular
query; but it would be far too expensive to consider a wide range of parameters
using a brute force method of simply trying many of them. Instead, we first
observe that moving the center a little bit tends not to affect the overall log-odds
very much. Thus, our search algorithm starts by trying centers on a coarse mesh.
It then selects the best one, and uses a finer grain mesh on the region around that
best one. This can be repeated until the desired accuracy is reached. In practice
we find that starting with points at every two degrees of latitude and longitude, and ending with points at tenths of degrees works well.

Once we have selected a center, we now have to optimize the other two parameters. Our approach is based on Theorem 5.2.3, below, which establishes that the log-likelihood as a function of $C$ and $\alpha$ is unimodal; we therefore develop techniques based on optimization of unimodal multivariate functions to find the optimal parameters. For scalability, we bucket all the queries by their distance from the center, enabling us to evaluate a particular choice of $C$ and $\alpha$ very quickly.

To establish the necessary unimodality property, we proceed as follows. Let $S$ be the set of log entries for query $q$ (indexed by users who issued this $q$), and let $d_i$ be the distance of a user $i$ from the query’s center. Then $f(C, \alpha) = \sum_{i \in S} \log C d_i^{-\alpha} + \sum_{i \notin S} \log (1 - C d_i^{-\alpha})$ is the log of the probability for parameters $C$ and $\alpha$.

**Lemma 5.2.1.** $f(C, \alpha)$ is concave in $C$.

**Lemma 5.2.2.** $f(C, \alpha)$ is concave in $\alpha$.

**Theorem 5.2.3.** $f(C, \alpha)$ has exactly one local maximum over its parameter space.

**Proof.** For sake of a contradiction, imagine that there were two choices of parameters, $(C_1, \alpha_1)$ and $(C_2, \alpha_2)$, each of which was a local maximum. Unless $\alpha_1 = \alpha_2$ (in which case they can’t both be maxima by Lemma 5.2.2), there is some $d_0$ such that $C_1 d_0^{-\alpha_1} = C_2 d_0^{-\alpha_2}$.

We now consider all functions $C d^{-\alpha}$ that cross the point $(d_0, C_1 d_0^{-\alpha_1} = C_2 d_0^{-\alpha_2})$. Each is fully determined by the value of the exponent $\alpha$, having $C(\alpha) = C_1 d_0^{\alpha - \alpha_1}$. We now consider the likelihood $f(C(\alpha), \alpha)$. By simply taking the second derivative of $f(C(\alpha), \alpha)$ with respect to $\alpha$, we find that it is always negative, contradicting
our original assumption of two maxima.

\[
f(C(\alpha), \alpha) = \sum_{i \in S} \log C_1 d_0^{-\alpha} d_i^{-\alpha} + \sum_{i \notin S} \log 1 - C_1 d_0^{-\alpha} d_i^{-\alpha}
\]

\[
f''(C(\alpha), \alpha) = \sum_{i \notin S} \frac{-(C_1 d_0^{-\alpha} (d_0^\alpha))^2}{(1 - C_1 d_0^{-\alpha} (d_0^\alpha))^2} + \frac{-C_1 d_0^{-\alpha} (d_0^\alpha) \log^2 \frac{d_0}{d}}{1 - C_1 d_0^{-\alpha} (d_0^\alpha)\beta}
\]

\(C_1 d_0^{-\alpha} (d_0^\alpha)\) is a probability in \([0, 1]\), which makes the first term at most 0. The second term is also non-positive for the same reason. The only way that the expression can evaluate to 0 is if \(C_1\) is 0. However, in this case the log-odds will have a log 0 for each \(i \in S\). \hfill \Box

In practice, our numerical methods converge quickly to the single maximum value. Our (unoptimized) implementation runs in under 30 seconds on a modern machine.

5.3 Assessing the Model

We now discuss some of the results that can obtained from this model, running on the complete set of query logs. These results can be organized by the kinds of queries for which spatial variation stands out: on the one hand, there are classes of queries that by their nature have a geographic focus (for example, names of schools, newspapers, or sports teams); and on the other hand, there are queries whose geographic content is a priori less apparent. Queries in this latter category can be found effectively by enumeration — that is, applying the model to all the
Figure 5.1: Geolocation of queries “Red Sox,” “Grand Canyon National Park,” and “Bell South”. (The capitalization of queries is reduced to a canonical form in our experiments.) These figures are drawn as heat maps, with the color spectrum indicating the query intensity per grid cell (and hence there is value in viewing these, as well as later figures, on a color display or color print-out). The arrows indicate the centers computed using our model.
frequent queries in the log and identifying those with large exponents, indicating geographic concentration.

We begin with the simplest kinds of examples — those queries for which there is a natural geographic focus. For analysis and evaluation, we consider several classes of such queries here: names of the most populous U.S. cities, names of certain universities, names of high-circulation U.S. newspapers, names of all Major League Baseball teams, names of all U.S. national parks, names of all U.S. Senators, as well as certain companies such as banks, airlines, and cell-phone carriers that have a regional focus. We will refer to these categories as our basic classes. Each query in one of the basic classes has an a priori natural geographic center, though the center may be conceptually a “point” (e.g. in the case of a national park or the home city of a sports team) or a broader region (e.g. in the case of a state represented by a Senator or a region served by a cell-phone carrier). In all cases, the model identifies these natural centers with high accuracy, as our more detailed evaluation below demonstrates.

By way of example, Figure 5.1 shows the query distribution for three queries in these classes: “Red Sox,” “Bell South,” “Grand Canyon National Park.” The first two queries are clearly easy to localize: We see that the query “Red Sox” has a conceptual hot-spot in New England, while the hot-spot for “Bell South” closely tracks the boundaries of the states that this company primarily serves. “Grand Canyon National Park” is one instance of a class of examples that is more subtle for several reasons. First, there is relatively little data on this query, even at the scale of complete logs; and as the image makes clear, the location of the park itself is found although the hot-spot is not immediately apparent visually. But beyond this, it should not have been clear in advance that the location of the park should
Table 5.1: Accuracy of algorithms for localizing senators inside their respective states.

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>Mean</th>
<th>Median</th>
<th>Local density</th>
<th>Our model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>51</td>
<td>12</td>
<td>80</td>
<td>90</td>
</tr>
</tbody>
</table>

even be a natural “center” for this query: the fact that it emerges as the center suggests that there is a hot-spot in query activity coming from people who are already at the physical location, rather than people from nearby population centers planning future vacations. We find this in general with geographic destinations in less-populated areas — despite the fact that a large fraction of visitors come from some distance away, the center is generally at the location itself.

In addition to the basic classes of queries, applying the model to all frequent queries turns up geographic information about queries that have no a priori “home.” As one illustration, the model uncovers the oft-discussed observation that different social-networking sites have particular concentration in different regions. For example, despite the enormous penetration of Facebook, a hot-spot is still uncovered in Southern Ontario — a fact about the Facebook user demographics that the company has remarked on in its own publicity. Similar observations are easily found for other social-networking sites as well.

We now turn from these examples to a more systematic evaluation of the model’s effectiveness at localizing centers.
5.3.1 Evaluation

We begin by formulating a framework within which to evaluate the model. The premise is that for a basic class in which the a priori natural centers have precise coordinates (e.g. the home city of a sports team), we define $a_q$ to be this natural center, and we define $b_q$ to be the center computed by the model. Evaluating the distance between them, $d(a_q, b_q)$, gives an indication of how accurate the model’s localization is.

To compare our model with simpler baselines, we also determine the distance from $a_q$ to centers computed by other means; specifically:

- $n_q$, the weighted center of gravity of all instances of query $q$;
Figure 5.3: Performance of algorithms on high-population U.S. cities.

- $m_q$, the point at the median latitude and median longitude of all instances of query $q$; and

- $\ell_q$, the point with the highest local density of instances of $q$ — that is, with the lowest likelihood relative to the overall base-rate for query $q$.

Note that the first two methods are geometric, while the third is probabilistic but much simpler than our model. In Figure 5.2, we compare all these methods at localizing all Major League Baseball team names to their home cities — in particular, depicting the cumulative distribution of distances to $a_q$ over all teams. We see that our model’s center $b_q$ and the optimum log-odds center $\ell_q$ greatly outperform the geometric methods, with both $b_q$ and $\ell_q$ localizing almost all team names to within 60 miles of their respective home cities. This is in a sense somewhat surprising,
given the multiple meanings (other than baseball-related ones) that many baseball team names have. Also, recall that our model, in addition to producing the center $b_q$, is also estimating dispersion in the form of the exponent $\alpha$, whereas one gets only a center from the baseline $\ell_q$. Due in part to the need to fit the full query distribution via this exponent, our model is less exact in its localization (compared to $\ell_q$) for distances significantly under 60 miles.

We perform an analogous evaluation for the names of all U.S. Senators, in which the natural center is no longer a point but a region (namely, their home state.) We evaluate, for our model and the three baseline methods, how many of the 100 Senators are localized to a center within the state they represent. Table 5.1 shows these results; our model outperforms all the baselines, with mean and median performing particularly poorly. (Certain queries in this class illustrate additional qualitative contrasts between the models; for example, our method localizes the query “Lisa Murkowski” to her home state of Alaska, while the three baseline methods all put the center in the continental U.S.)

It is also natural to evaluate our model against state-of-the-art commercial services, which employ features other than usage, for inferring whether a query is “local.” In particular, we use the service WhereOnEarth, a leading exemplar of this type of application. Our first finding is that query log data reveals strong spatial variation for much broader ranges of queries than services such as WhereOnEarth pick up. As a result, direct comparison is a bit difficult, since many of even our basic classes above are not considered localizable by these services. For example, WhereOnEarth does not consider the names of any U.S. Senators or Major League Baseball teams to be local queries for which a center can be inferred, despite the fact that our model finds correct centers for almost all from usage data.
Table 5.2: Estimation of exponents $\alpha$ for high-circulation U.S. newspapers.

<table>
<thead>
<tr>
<th>Newspaper</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Wall Street Journal</td>
<td>0.111327</td>
</tr>
<tr>
<td>USA Today</td>
<td>0.263173</td>
</tr>
<tr>
<td>The New York Times</td>
<td>0.304889</td>
</tr>
<tr>
<td>New York Post</td>
<td>0.459145</td>
</tr>
<tr>
<td>The Daily News</td>
<td>0.601810</td>
</tr>
<tr>
<td>Washington Post</td>
<td>0.719161</td>
</tr>
<tr>
<td>Los Angeles Times</td>
<td>0.782538</td>
</tr>
<tr>
<td>The Star Ledger</td>
<td>0.998462</td>
</tr>
<tr>
<td>Detroit Freepress</td>
<td>1.068055</td>
</tr>
<tr>
<td>San Francisco Chronicle</td>
<td>1.091030</td>
</tr>
<tr>
<td>Chicago Tribune</td>
<td>1.102554</td>
</tr>
<tr>
<td>Philadelphia Inquirer</td>
<td>1.140618</td>
</tr>
<tr>
<td>Chicago Sun Times</td>
<td>1.165482</td>
</tr>
<tr>
<td>The Boston Globe</td>
<td>1.171179</td>
</tr>
<tr>
<td>The Arizona Republic</td>
<td>1.284957</td>
</tr>
<tr>
<td>Dallas Morning News</td>
<td>1.286526</td>
</tr>
<tr>
<td>Houston Chronicle</td>
<td>1.289576</td>
</tr>
<tr>
<td>Star Tribune</td>
<td>1.337356</td>
</tr>
</tbody>
</table>

Table 5.3: Estimation of exponents $\alpha$ for the 10 most populous U.S. cities.

<table>
<thead>
<tr>
<th>City</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>New York</td>
<td>0.396527</td>
</tr>
<tr>
<td>Chicago</td>
<td>0.528589</td>
</tr>
<tr>
<td>Phoenix</td>
<td>0.551841</td>
</tr>
<tr>
<td>Dallas</td>
<td>0.588299</td>
</tr>
<tr>
<td>Houston</td>
<td>0.608562</td>
</tr>
<tr>
<td>Los Angeles</td>
<td>0.615746</td>
</tr>
<tr>
<td>San Antonio</td>
<td>0.763223</td>
</tr>
<tr>
<td>Philadelphia</td>
<td>0.783850</td>
</tr>
<tr>
<td>Detroit</td>
<td>0.786158</td>
</tr>
<tr>
<td>San Jose</td>
<td>0.850962</td>
</tr>
</tbody>
</table>
Table 5.4: Estimation of exponents $\alpha$ for the 10 highest-ranked U.S. universities according to U.S. News & World Report.

<table>
<thead>
<tr>
<th>School</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harvard</td>
<td>0.386832</td>
</tr>
<tr>
<td>Caltech</td>
<td>0.423631</td>
</tr>
<tr>
<td>Columbia</td>
<td>0.441880</td>
</tr>
<tr>
<td>MIT</td>
<td>0.457628</td>
</tr>
<tr>
<td>Princeton</td>
<td>0.497590</td>
</tr>
<tr>
<td>Yale</td>
<td>0.514267</td>
</tr>
<tr>
<td>Stanford</td>
<td>0.627069</td>
</tr>
<tr>
<td>U. Penn</td>
<td>0.729556</td>
</tr>
<tr>
<td>Duke</td>
<td>0.741114</td>
</tr>
<tr>
<td>U. Chicago</td>
<td>1.097012</td>
</tr>
</tbody>
</table>

For other basic classes, such as high-population U.S. cities, major U.S. universities, and U.S. national parks, WhereOnEarth can determine exact values for almost all by table look-up. Our model does well in all these cases too, despite having no comparable access to hard-coded data; and it outperforms the three baselines $n_q$, $m_q$, $\ell_q$ in all these cases. Figure 5.3 shows the performance for U.S. cities; note that our model significantly outperforms the other three approaches, and the the center of gravity $n_q$ outperforms the simple probabilistic baseline $\ell_q$ in this case.

### 5.3.2 Exponents and Dispersion

Thus far we have been considering the centers computed by the model, but there is additional value in the exponent as well. This provides the measure of dispersion mentioned in the introduction; a large exponent indicates rapid decay away from the center, and hence strong geographic concentration, while a smaller exponent indicates interest over a broader region.
Thus, particularly when we compare exponents for queries from the same basic class, we can place items on a spectrum ranging from local appeal to more national appeal. For example, Tables 5.2-5.4 show the exponents for the 10 most populous U.S. cities, for the 10 highest-ranked U.S. universities according to U.S. News & World Report, and for a collection of high-circulation U.S. newspapers.

Ranking each of these lists by exponent places them on a spectrum from local to national appeal. For example, the Wall Street Journal and USA Today are the two newspapers with the lowest exponents, indicating national interest, with the New York Times close behind. Other high-circulation newspapers are regional in their appeal, with exponents that are much higher. We also see that the spatial variation in queries for city names does not directly correspond to the populations of the cities; for example, Los Angeles has a comparatively large exponent, while the second-lowest exponent among large U.S. cities belongs to one that is not in the top 10: Las Vegas, with an exponent of .482. While we omit the list of national parks due to space limitations, there is significant variation in exponents here too, with Death Valley, the Grand Canyon, and the Everglades having the lowest values (and hence the most national reach in queries).

5.4 Extensions: Temporal Variation and Multiple Centers

5.4.1 Temporal Aspects

While most localizable queries maintain relatively stable centers and dispersions over time, it is easy to find queries which vary in both of these dimensions. A local news story might start with limited dispersion as only people in the region
are aware of it. If the story then gains national attention, the center may stay
the same, but the exponent $\alpha$ can decrease as query traffic increases from farther
away.

In other cases, the center may move as well, and a good source of examples for
this comes from large weather phenomena. For instance, as a hurricane moves over
time, the people who are next in its path at any given moment tend to search for
it with the highest intensity, and thus we might expect the query center to roughly
track the storm’s center.

We can observe this in the query-log data by considering a sequence of 24-
hour time slices, at offsets of one hour from each other (i.e. considering the 24
hours starting at midnight, then the 24 hours starting at 1 AM, and so forth).
Figure 5.5: Change in the exponent for “Hurricane Dean” by hour, as interest in the topic shifted from local to national.

We can then calculate the center for each of these 24 hour periods. By using a sliding window in this way, we are able to fit our parameters to more data, making them more reliable. Employing 24-hour periods has the useful effect of mitigating diurnal variation, since all times of day are represented in each period.

In the case of a major recent hurricane, Hurricane Dean, we observe a clear westerly trend, with centers starting far out in the Caribbean, and tending to move westward towards Southeast Texas as the hurricane does so. There is also a clear trend towards decreasing $\alpha$ as the hurricane gains national attention.

For a number of reasons, however, the sequence of query centers computed for each time slice in isolation does not move very smoothly. In large part, this is
because the amount of query-log data for a single hurricane, even a major one, is relatively small, especially before it approaches mainland North America. Thus, we improve the results using a more complex method: we couple the computation of centers across the different time-slices by a natural and efficient algorithm, obtaining a smooth query-center path that tracks the true path of the storm’s center with surprising fidelity (Figure 5.4).

The coupled computation works as follows. For each 24-hour period, and each latitude and longitude, we compute the cost of the center at that location as the negative log-probability for the optimal choice of $C$ and $\alpha$. In order to account for difference in total volume between different 24-hour periods, we normalize by dividing the cost at a point $A$ by the the minimum cost over all possible centers for that same 24-hour period. Thus, we have a normalized cost for every coordinate for every time window. We now define a cost for moving the center from point...
Figure 5.7: Distinctive queries for locations in the San Francisco Bay Area.

A to point B as $\gamma |A - B|^2$. Thus paths which jump around a lot are penalized, while smooth paths which move at a constant rate have relatively low cost. The goal now is to find a sequence of centers for the sequence of 24-hour windows (each offset by one hour from the previous one) that minimizes the sum of the costs from the placement of the centers and the costs for the movement from each center to the next.

It is easy to find the lowest-cost sequence of centers for various constants $\gamma$ using dynamic programming. Once we have done this, we can examine the smoothed paths taken by the center of the query for different $\gamma$, one of which is shown in Figure 5.4. We see a striking similarity between the actual path of the storm’s center and the smoothed path taken by the computed center.

In addition to tracking the storm’s motion through query logs, we can also watch how the query’s dispersion changes over time (Figure 5.5). By examining
the optimal choices of $\alpha$ over time for the smoothed path, we can see that the hurricane started out as an event of very local interest, with its center near the Lesser Antilles. As the storm moved west and intensified, more and more people started taking notice, and it eventually became a major news story, as it was the one of the most intense hurricanes to ever reach land.

5.4.2 Multiple Centers

While the simple, single-center model describes many queries fairly well, some queries are clearly better modeled with multiple centers. For example, major airlines typically have three or four hub cities, and it is clear from the query-log data that the regions around each of these cities have high query frequency for their respective airlines.
To model this sort of spatial variation we extend our generative model by placing multiple centers, each with its own $C$ and $\alpha$ parameters. For each point, we use the probability given by the center which yields the highest probability to that point. Thus, we can easily calculate the log-odds for a choice of multiple centers with different parameters.

This, however, makes the maximum-likelihood optimization problem much harder, and so we use a heuristic based on the $K$-means clustering algorithm. We start with $K$ centers placed at random, for some constant $K$. We then optimize each of these centers, treating each one as if it were the only center being used (our previous algorithm). After we do this for every center, we look at each geographic point and determine which of the $K$ centers gives that point the highest probability, according to the polynomial-decay probability function. We now say that each point is associated with the center giving it the highest probability. We then reoptimize each center independently, but considering only those points that were associated with it during the previous iteration. This process is repeated a number of times until it converges. The algorithm is sensitive to the starting locations of the centers, but by running it many times and choosing the best outcome, we achieve good results.

As an illustration, Figure 5.6 shows the results of this algorithm for the query “United Airlines.” United’s largest hub is in Chicago, and it also has hubs in Denver, Washington DC, San Francisco, and Los Angeles. The algorithm places centers in Chicago, Washington, and near Denver. It places a fourth center off the coast of California, which has the effect of hitting both San Francisco and Los Angeles somewhat equally. (Note that it is a natural consequence of the probabilistic model, even with one center, that there may be low query density at
the exact point corresponding to the center itself.) We see similar results for other airlines.

The multiple-center model is also useful for queries with two distinct geographic meanings. For example, on the query “Washington,” with two centers, the algorithm places one center in DC and the other in Washington state. For the query “Cardinals,” the algorithm places one center in St. Louis (the home of the baseball team) and the other in Arizona (the home of the football team).

5.5 Enumerating Multiple Queries on a Shared Map

5.5.1 Distinctive Queries for all Locations

Given a way to assess the spatial variation of individual queries, we can enumerate all the queries in the log and — for each location on earth — find the queries that are the most “unusual” or “distinctive” for that location. In this section we describe the results of such a computation, leading to an annotated world map of which the image in Figure 5.7 is a tiny portion.

We define locations as before, using tenth-of-a-degree grid cells. (For reference, such a cell has a side length of less than 10 miles.) We define “distinctiveness” at a given cell \( x \) using a variant of the probabilistic model from Section 2. For each query \( q \), let \( p \) denote the fraction of all entries in the log corresponding to users issuing \( q \). Let \( t_x \) be the total number of log entries from \( x \), and let \( s_x \) be the number of log entries from \( x \) corresponding to users issuing \( q \). Assuming a simple independence-based model, the probability of this observed data given the
background probability $p$ is $(t_x)^p (1 - p)^{t_x - s_x}$. We choose the queries $q$ for which this probability is lowest to serve as the most distinctive queries for location $x$ — they are the ones that deviate the most significantly at $x$ from their global background rate.

We perform the computation on queries issued during a week. To have sufficient sample size, we only consider locations with at least 5000 queries during the week. This yields 3154 locations worldwide with 2643 locations in the continental US; for each, we find the most distinctive queries.

We illustrate some of the results of this in Figure 5.7; for ease of presentation, we only display the San Francisco Bay Area, and only a subset of the locations there. While one might have expected a region this small to have roughly the same distinctive queries in each cell (things like “San Francisco,” “San Jose,” and so forth), in fact we see significant and meaningful differences between locations that are only a few miles apart (see for example, the distinction between queries being issued in Palo Alto and Sunnyvale.)

### 5.5.2 Spheres of Influence

If we consider many of the basic classes of queries from Section 3, they represent entities that are at least implicitly in competition. (Consider, for example, baseball teams, universities, or newspapers.) By representing all the queries from a single class on a shared map, we can try understanding and visualizing their respective “spheres of influence” — the regions in which each is dominant. In Figure 5.8, we depict such regions of influence for all Major League Baseball teams on a map of the U.S.: thus the team names are the queries, and each grid cell is colored
according to the distribution of queries for baseball teams issued from that cell. We now discuss the methodology underlying images such as this, then make some observations about this image itself.

To define regions of influence, we need a way to represent the queries that are dominant at different grid cells. The simplest way would be just to see which query produces the most log entries for each cell, and color the cell with this most abundant query. However, due to the sparsity of the data, this produces a fairly noisy representation, with adjacent cells generally differing in color, and it doesn’t capture the difference between minor dominance and strong dominance in a given cell.

A better way to represent the regions of influence, then, is to imagine that each query from a particular cell acts as a vote for that query. The pixels are then colored by blending colors according to the voting. Done in a straightforward way, this now has the opposite problem — most regions are too blended in the image. To strike a balance, we produce the image in Figure 5.8 by counting a query with \( N \) log entries in a cell as having \( N^c \) votes for a small constant \( c > 1 \). Varying \( c \) lets us bring out the dominant color, but still allowing blending when there are close competitors.

The first observation about Figure 5.8, of course, is that despite the highly uneven national volumes of queries for different teams, we see a very clean geographic breakdown of who follows which teams. It is also somewhat surprising to see the extent to which team dominance breaks down along state lines — a number of the colored regions follow state boundaries very clearly. For instance, in Michigan, across the lake from Chicago but far from Detroit, it is the Tigers, not the Cubs who have the largest following. It is also interesting to note the regions which do
not have a clear-cut winner. For instance, in the Carolinas and Louisiana there are many queries for baseball teams, but there is no one team that stands out above the rest.

### 5.6 Discussion

We have seen that large-scale query-log data contains enough information to build effective models of spatial variation. In addition to finding centers for queries with an accuracy that outperforms competing baselines, and extracting geographic information for a broader range of queries than is accessible to commercial systems, our models also form the basis for algorithms that incorporate temporal processes, as well as methods to analyze variation for many queries simultaneously at a global level.

There are a number of directions in which this work could be extended. It would be interesting to consider our analysis of simultaneous spatial and temporal variation (as in Section 4.1) in the context of further probabilistic models, potentially exploring connections with the methodology in [94]. It would also be interesting to incorporate more complex models of user behavior into a framework that explicitly took spatial variation into account, potentially resulting in more accurate kinds of localization for broader classes of queries. Ultimately, as the local applications of search continue to broaden, we can expect to see questions of this sort arise increasingly from the rich interaction between Web information, user interests, and the geographic and spatial frames of reference in which they are embedded.
5.7 Related work

Prior work related to this chapter can be grouped roughly into four high-level areas: geolocation of Web content, geolocation of search queries, efficient query processing with geographic information, and spatial hot-spot models.

There is a significant line of work on inferring geographic locations for Web pages and other on-line content. Buyukkokten et al. [28] use geographic entities and network IP address information to geolocate Web pages. McCurley [93] proposed a spatial browsing of Web data; his approach was to use a rich set of geographic features, including telephone numbers, to infer geospatial context. Amitay et al. [9] describe a system, Web-a-Where, that assigns each page a geographic focus. Further work includes [92, 96, 126]. Ding et al. [40] introduced the idea of the power and spread of a Web page, which are analogous to the center and dispersion parameters in our model. However, their approach is not based on a probabilistic model or corresponding optimization criterion for the parameters. Some applications of Web content geolocation include mining spatio-temporal themes [94] and geographically focused collaborative crawling [58].

In contrast, much less work has been done on geolocating Web queries, our focus here. Gravano et al. [63] performed an early investigation of this issue, using machine learning techniques to classify search queries as either local or global. Closer to our work, the paper of Wang et al. [125] searches for the “dominant location” of a query in the context of a system for exploiting query localization to improve retrieval performance. They include power and spread among their features, and again the approach is quite different, and does not include a specific model for spatial variation.
Figure 5.9: Automatically generated landmark map of New York City. The maps itself simply shows the geotagged image density.

Query processing with geographic constraints is an active research area. Much work here constructs and processes spatial representations to enable efficient query processing. Some recent work in this area is by Chen et al. [31], Tezuka et al. [120], and Schockaert and De Cock [112].

Finally, spatial hot-spot models have been extensively studied in statistics. For a good account of the work in this area, ranging from the ad hoc to the model-
Figure 5.10: Automatically generated landmark map of London. The map itself simply shows the geotagged image density.

based, see the discussion in Neill et al. [99], as well as Kulldorff [78] and the book edited by Lawson and Denison [84]. There are also connections to temporal hot-spot models, which use one-dimensional analogues of these computations [45, 72]. In work subsequent to our study of search engine queries here, we looked at similar types of geographic information in the context of the photo-sharing site Flickr [35]. Here we showed how to use geotagged photographs on Flickr to automatically identify geographic hotspots corresponding to popular landmarks, and how to combine these with photographic data to generate automatically annotated landmark maps. Figure 5.9 and Figure 5.10 show the results of this method in New York City and London.
CHAPTER 6

OPTIMIZING WEB TRAFFIC VIA THE MEDIA SCHEDULING PROBLEM

Many websites have featured items placed prominently on their web pages. News sites have featured news stories, content sharing sites (like Flickr and YouTube) have featured media, and on-line stores have featured products. The exact business goals of these sites vary, but in general these items are placed in featured locations with the expectation that a large fraction of visitors will examine them. For a news site, the featured article is typically a story that many people will be interested in, and one metric of success that can be used by a news site when evaluating its selection of featured articles is the total number of clicks. A news site that does a good job of delivering the news that is interesting to most people will have a large number of overall clicks. Similarly, a content-sharing site would like to present high-quality content in featured spots. An on-line store might have slightly different goals in selecting featured products: they might select high-margin rather than popular items, for instance.

In all of these cases, the website operator sees some value in having the featured items appear prominently and receiving user attention. Some utility is gained for each impression of the featured item. The visitor may think more highly of the website the featured item is placed on, or be inspired to follow a link to another page, or even buy a product. In this work, we will consider the value of an impression to be represented by the probability of a user clicking on that featured item, but other tasks, such as maximizing total sales or total profit can be considered within our framework as well.
6.1 The Media Scheduling Problem

There is thus an underlying content scheduling problem that is central to the choice of featured items on all such sites. However, this scheduling problem has remained essentially implicit and unexamined in the literature, despite its role in determining the clickthrough rates for content that is in some cases viewed by hundreds of millions of individuals. In this chapter we provide a formulation of this problem in terms of a few underlying parameters that can be reliably inferred from a site's log data; and by formulating the problem precisely, we are able to develop algorithms that improve significantly on current practice. While the problem in its most general form is intractable, we provide efficient algorithms for a special case that closely approximates data from real user traffic. We then evaluate our algorithm on trace data from the front page of Yahoo! (the most visited Web site in the world), showing how to improve the estimated clickthrough rate on featured articles by over 25% relative to the human-intensive strategies that are currently employed.

We now give an overview of the problem, before formally specifying it in the next section. The operator of a website has a pool of potential items that can be featured on the front page over the course of a day. We assume this pool is known at the start of the day: this is appropriate for most applications, including advertising, general-interest news features as on the Yahoo! front page, highlighted videos and photos as on the YouTube and Flickr front pages, and a number of other typical settings.\footnote{The main setting in which it is less appropriate is in the handling of breaking news, which can come unexpectedly and is time-critical. In the final section we discuss some interesting directions for future research based on adapting our model to this setting.}

User interest in a single featured item decays over time as it remains on a site;
this is largely due to the fact that the user population viewing it will increasingly consist of repeat visitors who have seen it linked from the front page already. Thus, as we show in Section 6.3, user interest in a featured item can be characterized by two parameters: a peak interest level, which is the probability of a user clicking on the item when it is first presented; and a decay function, which is the item-specific rate over time at which this click probability decreases while the item is being featured. Moreover, these parameters can be quickly estimated using bucket tests on a small subset of the full user population. In Section 6.3 we show that the website operator can also reliably estimate a final important parameter in the formulation of the problem: the user traffic per minute over the course of a day.

We thus have the following media scheduling problem, which will be the focus of the paper: given the estimated time series of user traffic for the day, and a pool of potential items to be featured — each with its own peak interest level and decay function — determine an order in which to feature items, and a length of time that each should be featured. This formulation of the problem captures some of the fundamental trade-offs at work in scheduling items for high-volume sites. In particular if we always present the best item, repeat visitors will have no opportunity to see other items, and the value per impression will decrease as more and more of the impressions are from repeat visitors. If we change the items too often, we will not extract maximum value from the best items, as low quality items will often fill the featured spot.

6.1.1 Our Results for Media Scheduling

Since the general version of the problem is NP-hard, we seek tractable cases that approximate real traffic data. This is challenging, since not only does user traffic
vary considerably over a day and peak interest vary considerably from one item to another — even the decay functions of different items can have quite different shapes. Using full traffic data from the Yahoo! front page, however, we are able to identify a crucial phenomenon in the data that leads to a tractable formulation: the decay functions for different items can all be approximately fit to (different) segments of a single “universal” curve. We then show that for any instance of the media scheduling in which decay functions all form segments of a single curve, and in which traffic over the course of a day is unimodal (or bimodal), the media scheduling problem can be solved efficiently. When these properties hold to within some approximation bound — as they do in our case, with small error — our algorithm provides the same approximation guarantee to the optimum schedule. We evaluate the algorithm in comparison both to simpler baselines and to the way content is actually scheduled on the Yahoo! home page, showing significant gains in total clicks. Thus, this is a case where the analysis of a large volume of data, and the identification of regular structure in it, feeds directly into algorithm design (and corresponding performance improvements) for a large-scale application.

The problem formulation, as argued above, is general enough to apply to a wide range of high-traffic sites. We believe that our observations about the data will be useful in the context of many other sites as well, since they are aggregates of hundreds of millions of visitors and do not appear to depend on any idiosyncrasies of Yahoo!’s content presentation. With this in mind, we explore the structure of the traffic data in detail, identifying principles about clickthroughs and decay rates, as well as a simple generative model that explains some of the fundamental patterns that we see.
6.2 The Problem Formalized

We now formalize the media scheduling problem. We have $N$ items that we could potentially place in the featured location. Each item has an associated value function $f_i(t)$, which specifies the utility of a visiting user seeing item $i$ after it has been on the front page for $t$ minutes. In our applications, we will think of $f_i(t)$ as giving the probability each user clicks on the item, when visiting the page $t$ minutes after the item’s first appearance. However, in general, $f_i(t)$ can be any measure of utility per visitor and the formulation remains the same. Based on traffic patterns from previous days, we also know with high accuracy how many people will show up to the front page each minute of the day: $a_\tau$ at minute $\tau$. Our goal is to determine which subset of the items to use, when to place them, and for how long, in order to maximize the total value over the course of the day.

Thus, the problem is to assign non-overlapping intervals $[S_i, T_i)$ to each item $i$, where the expected number of clicks on item $i$ at time $\tau \in [S_i, T_i)$ is $a_\tau f_i(\tau - S_i)$. (This choice includes selecting a subset of the items to show, since we can choose not to use an item by setting $S_i = T_i$.) Our goal is to maximize $\sum_i \sum_{\tau = S_i}^{T_i} a_\tau f_i(\tau - S_i)$.

Recall that we are considering applications in which each piece of content can be shown at any point during the day. Also, the fact that each item has a function $f_i(t)$ indicates that the value per visitor is assumed to be independent of the time of day.\footnote{An interesting extension is to explicitly consider the fact that some content may be more appealing to typical visitors at, say 6 AM, than at 6 PM. We do not have indications that this is a strong effect for the kinds of applications (i.e. general-interest Yahoo! news items) where we have data for analysis; but there are certainly settings where this can be an effect. We indicate it as a direction for future work.} Additionally, we are constrained to put each item up for at most one contiguous interval: once it is replaced, it cannot return. This is consistent with
practice on most of the websites we consider, where human editors tend not to bring back a featured item later, since it creates for the user population an unwanted impression of non-fresh, recycled content on the site.

6.3 Data and Temporal patterns

To optimize the media scheduling problem effectively, we need to be able to predict the overall traffic ($a_t$) and item clickthrough rates ($f_i(t)$) ahead of time. To illustrate the feasibility of estimating such data effectively, we examine the server logs for the front page of Yahoo!

Figure 6.1 gives an example showing the typical placement and positioning of the featured articles which we examine on the Yahoo! front page. (“Best and worst French fries” was the article being featured at the time this image was captured.) Over the course of three weeks (starting May 1, 2008), we recorded the number of front page views and the number of article clicks at a 1-minute resolution. Using
Figure 6.2: The number of page views of yahoo.com over the course of three consecutive weeks in Nov/Dec 2008. Note, that with the exception of a small glitch (likely cause by a logging error or server failure) the traffic is very consistent over the three weeks.

this data, we are able to ascertain the clickthrough rate for an article after it has been featured on the website for some number of minutes. In this study we will only consider articles that were featured for at least 20 minutes between the hours of 3AM and 8PM Pacific time. This gave us a total of 337 featured articles, for an average of about 16 per day. While our algorithm could have used all articles (not just those shown for at least 20 minutes) and could have scheduled the full 24-hour day, the optimal algorithm to which we compare starts to become prohibitively slow as the number of articles exceeds 20. Hence we imposed these restrictions, both for our algorithm and competing approaches, to allow comparison to the optimum without removing very many high-quality articles, or high traffic times of day.

Figure 6.2 shows the total number of views of *yahoo.com* as a function of time for three weeks. Over these three weeks, the view curves are almost identical — e.g.,
4:53 on one Wednesday has about the same number of views as 4:53 on another Wednesday. This illustrates the predictability of traffic on most days: during regular weeks, the traffic pattern is almost identical from one week to the next. We note that on some days, particularly holidays, the traffic does not conform to the typical pattern.\(^3\)

Predicting the time-varying clickthrough rates of media content is a harder task. However, we can use bucket testing to get a good approximation of this. By presenting a small fraction of the visitors to our site with a particular piece of content, we can get a good approximation of how popular that piece of content will be when presented to the entire population. Once we have learned the initial popularity of an article, the future popularity will follow a predictable decay. (See

\(^3\)To predict the traffic on holidays, we would have to use a slightly more sophisticated model, perhaps looking back a year to that day last year, or perhaps estimating from similar holidays.)
[4] for more sophisticated statistical models for predicting clickthrough rates in the context of content recommendation.)

Crucially for our analysis, we find that these articles not only follow predictable decay curves, but they can all be approximately realized as different segments of a single universal curve, as illustrated in Figure 6.3. That is, there is a single curve $g(t)$, so that each item’s clickthrough function $f_i(t)$ can be written as $f_i(t) = g(t + \sigma_i)$ for an item-specific offset $\sigma_i$. Thus, when learning the parameters for an article $i$, we need only use bucket testing to learn the appropriate starting point $\sigma_i$ along this single curve. If, for instance, we find that the initial clickthrough rate (when the article is first featured) is 5%, we can find the point along this single curve corresponding to a clickthrough rate of 5%. We find that this corresponds to some $x = T$ on the plot in Figure 6.3. Once we know this offset, we can compute that the clickthrough rate after the article has been up for 20 minutes will correspond to $T + 20$ in Figure 6.3. Thus with only the initial rate, we can accurately predict the clickthrough rates we will achieve when placing this article in a featured position for any length of time. In our data, we observe that all articles can be aligned to a single monotonically decreasing function such that the average relative error is only 3.2%.

A natural worry is that with only a few hundred articles in our dataset, we are overfitting the data and cannot, in reality, predict $f_i(t)$ for all $t$ from the first few minutes of data and a small sample population. To show that this is not the case, we also fit the data to a simple exponential decay curve. While each article has its own starting point, there is only one universal parameter to this function: the decay parameter $\lambda$. We find that the best fit is when $\lambda = -0.0044$, indicating that the clickthrough rate typically declines by 0.44% per minute in our data. While
Figure 6.4: A fit of three functions to the single longest running article in our dataset. The exponential decay curve is clearly the best, and while the linear curve isn’t too far enough, it will certainly continue to deviate more and more from reality since it will eventually become negative.

This fit is not as good as the previous one (it can’t be since only the constraint on the previous curve was that it be decreasing) it still gives an average relative error of only 4.6%.

Figure 6.4 shows the fit of this exponential decay for the single article which appears for the longest interval in our data. While it does not fit quite as well as one might hope, it is a good approximation, and is significantly better than another simple alternative: a power law decay. For comparison, the average relative error of the exponential fit is 4.1%, while the power-law and linear fits have mean relative error 13.1% and 8.2%, and the best constant function with fixed $y$-value would have average relative error of 30.3%. In other words, these clickthrough rates change significantly over time, and while an exponential decay is not perfect, it gives a close approximation, better than other comparably simple functions.

In Section 6.6 we will investigate the declining clickthrough rate in greater
detail and offer an explanation for this phenomenon. For now, however, it suf-
fices to observe that the functions \( f_i(t) \) are reasonably predictable, and can all be approximately aligned to segments of one common curve.

### 6.4 Algorithms

In this section, we describe the development of our algorithms. As a preliminary step, we show that the general problem is NP-hard, but that even in general, there is an exponential algorithm that improves significantly over brute-force search (making it possible to find optimal solutions for \( N \) around 20-30 rather than 10). As our main focus, we then show how to solve the problem in polynomial time when the functions \( f_i(t) \) can all be aligned to segments of a single universal curve — as we observed for our data in the previous section — and when the user traffic is unimodal (or \( K \)-modal for any fixed \( K \)) over the course of the day. When these conditions are approximately met, our algorithm here produces a corresponding approximation guarantee to the optimum.

#### 6.4.1 Preliminaries

**NP-Hardness**

**Theorem 6.4.1.** The general media scheduling problem is NP-hard.

*Sketch.* A reduction from bin-packing gives \( K \) disjoint intervals where traffic is 1 and the interval lengths are equal to the bin size. These intervals with traffic 1 are separated by intervals of the same size with zero traffic. We reduce the items in the
bin-packing problem to items whose clickthrough rates are 1 up to time equal to
the length of the corresponding bin-packing item, and 0 thereafter, optimal clicks
can be achieved if and only if the original bin-packing problem is solvable.

Optimal Algorithm  A naive algorithm would have to consider $N!$ different
permutations of the items, and even then it would still have to determine the
interval lengths for each one. This would make it infeasible to solve instances
even of size 15. Using a more sophisticated algorithm considerably improves the
exponential behavior of the search for the optimum.

**Theorem 6.4.2.** The optimal solution for any traffic pattern $a$ and any functions
$f_i$ can be found in time $O(T^2N2^N)$ where $T$ is the total time, and $N$ is the number
of media items in the set of all available items.

*Proof.* An algorithm using dynamic programming can achieve this bound. For
each $t \leq T$ and each subset $S$ of items, the algorithm computes the optimal
scheduling for the first $t$ minutes of the day, using only the items in $S$. To do
this, we note that the optimal scheduling for some choice of $t$ and some subset
$S$ places a particular item last for some time interval from $t - u$ to $t$. Call this
item $i$. Then, the value of the optimal scheduling for this choice of $i$ and $S$ is
$\text{opt}(t - u, S \setminus \{i\}) + \text{value}(t - u, t, i)$, where $\text{value}(a, b, i)$ is the value of placing item
$i$ in the interval $[a, b)$ and can easily be precomputed. Of course, we do not know
which $u$ and $i$ to use ahead of time, but by searching over all choices of $u$ and $i$,
we can find $\text{opt}(t, S)$. There are $O(T2^N)$ choices for $t$ and $S$, and computing each
one requires searching over $O(TN)$ choices for $u$ and $i$. Multiplying these together
gives us the runtime stated. $\square$
Figure 6.5: Illustration of the proof of Lemma 6.4.3. If we have a situation where the ‘better’ (blue) item is placed earlier, we can swap the two items as illustrated here. After this swap, the clickthrough rate remains unchanged in the last $\gamma$, while it drops in the lower traffic region from 0 to $\delta$, and rises in the higher traffic region from $\delta$ to $\delta + \sigma_j$.

6.4.2 Our Algorithm

As we observed in Section 6.3, the clickthrough rates for all the items can, to a close approximation, be aligned to a single curve. Formally, if the clickthrough rate for item $i$ after $t$ minutes is given by $f_i(t)$, we can say that there is some universal, monotonically decreasing function $g(t)$ such that $f_i(t) = g(t + \sigma_i)$ for some $\sigma_i$.

Furthermore, as shown in Figure 6.2, the traffic rate over the course of a single day is approximately unimodal: it increases in the mornings, peaks, and then decreases in the afternoons. (It is even closer to being bimodal, with a dip at midday, and we can handle this as well.) The crucial building block in handling unimodal traffic (or $K$-modal traffic for fixed $K$) is to understand the problem when traffic is monotonically increasing or decreasing. We do this first, via the following two lemmas.

**Lemma 6.4.3.** *If the traffic is monotonically increasing, then an optimal ordering of items is given by decreasing $\sigma_i$, i.e., putting worse items earlier.*
Proof. Suppose there were an optimal ordering where item $i$ comes immediately before item $j$ in the optimal ordering, but $\sigma_i < \sigma_j$, in contrast to the statement of our lemma. We will show how this leads to a contradiction. Without loss of generality, we will assume that $\sigma_i = 0$. This optimal solution presents item $i$ for $\sigma_j + \delta$ minutes, followed by item $j$ for $\gamma$ minutes. An optimal solution must have $\delta \geq 0$, since if $\delta < 0$, then $f_i(\sigma_j + \delta) < f_i(\sigma_j) = g(\sigma_j) = f_j(0)$. In other words, if $\delta < 0$, then $f_i$ at the end of item $i$’s interval would be greater than $f_j$ at the beginning of item $j$’s interval, and we could do better by extending item $i$ and starting item $j$ a little bit later.

We now consider what would happen if we swapped the order of the two items, presenting item $j$ first for time $\delta$ and item $i$ for time $\sigma_j + \gamma$. This is illustrated in Figure 6.5.

These two items span a total time of $\delta + \sigma_j + \gamma$. When item $i$ came first, the total clicks were

$$\int_0^{\sigma_j+\delta} a_x f_i(x) \, dx + \int_{\sigma_j+\delta}^{\sigma_j+\delta+\gamma} a_x f_j(x-\sigma_j-\delta) \, dx.$$

When we perform the swap suggested above, the total clicks become

$$\int_0^\delta a_x f_j(x) \, dx + \int_{\delta}^{\sigma_j+\delta+\gamma} a_x f_i(x-\delta) \, dx.$$

For time greater than $\sigma_j + \delta$, the total click contribution is unchanged. In this interval, the optimal ordering had a clickthrough rate of $f_j(x-\sigma_j-\delta) = g(x-\delta)$, while our swap gives a clickthrough rate of $f_i(x-\delta) = g(x-\delta)$ — the same. We now consider two cases:
Case 1: \( \sigma_j < \delta \).

**Loss** Because of the swap we make from the optimal solution, the clickthrough rate decreases in the interval from 0 to \( \delta \) as \( f_j(x) = g(x + \sigma_j) < g(x) = f_i(x) \). The decrease in this region can be written as
\[
\int_0^{\delta} a_x(f_i(x) - f_j(x)) \, dx = \int_0^{\delta} a_x(g(x) - g(x + \sigma_j)) \, dx.
\]
Because \( a \) is monotonically increasing and \( g \) is monotonically decreasing we can bound this loss in clicks by \( a_\delta \int_0^\delta g(x) - g(x + \sigma_j) \, dx \). Because \( \sigma_j < \delta \) this telescopes and can be rewritten as \( a_\delta \int_{\delta}^{\sigma_j} g(x) - g(x + \delta) \).

**Gain** On the other hand, in the interval from \( \delta \) to \( \sigma_j + \delta \), the clickthrough rate increases, as \( f_i(x - \delta) > f_i(x) \). Here, the total gain we get is \( \int_{\delta}^{\sigma_j + \delta} a_x(f_i(x - \delta) - f_i(x)) \, dx \). Since \( a \) is increasing, this is at least
\[
a_\delta \int_{\delta}^{\sigma_j + \delta} (f_i(x - \delta) - f_i(x)) \, dx = a_\delta \int_{\delta}^{\sigma_j} (g(x) - g(x + \delta)) \, dx.
\]
Thus, the loss in clicks in the interval from 0 to \( \delta \) is at least offset by the gain we achieve in the interval from \( \delta \) to \( \sigma_j + \delta \) when we make the swap from the optimal ordering, and we do no worse (strictly better if the monotone functions are strict) from reordering items \( i \) and \( j \).

Case 2: \( \sigma_j \geq \delta \).

The analysis here is similar except we collect terms by \( g \) instead of \( a \). The loss for clickthrough rates between \( g(\sigma_j) \) and \( g(\sigma_j + \delta) \) can be written as
\[
\int_{\sigma_j}^{\sigma_j + \delta} g(x)(a_x - a_{x-\sigma_j}) \, dx \leq g(\sigma_j) \int_0^{\sigma_j} (a_{x+\delta} - a_x) \, dx.
\]
The gain for clickthrough rates between \( g(0) \) and \( g(\sigma_j) \), on the other hand, is
\[
\int_0^{\sigma_j} g(x)(a_{x+\delta} - a_x) \, dx > g(\sigma_j) \int_0^{\sigma_j} (a_{x+\delta} - a_x) \, dx.
\]
As before, the gain from the swap is at least as large as the loss.

Lemma 6.4.4. If the traffic is monotonically decreasing, then an optimal ordering of items is given by increasing $\sigma_i$, i.e., putting better items earlier.

Proof. While this seems similar to the case in Lemma 6.4.3, we note that it is more than just a case of changing the signs and performing the same analysis. In the previous case, the traffic was increasing through time, while the clickthrough rate was decreasing. In this case, however, the traffic is decreasing as well as the clickthrough rate. Thus, while similar, the analysis ends up being a bit different, and hence this lemma needs an argument separate from Lemma 6.4.3.

Consider an optimal ordering where this is not the case: item $j$ proceeds item $i$, but $\sigma_j > \sigma_i$. As in the proof of the previous lemma, we will show how this leads to a contradiction. Without loss of generality we will assume that $\sigma_i = 0$. We will denote the length of the interval for item $i$ as $\sigma_j + \delta$ and the length of the interval for item $j$ as $\gamma$. If we find that $\delta < 0$ here, we decrease $\sigma_j$ to the point where $\delta = 0$. This would only increase the quality of the optimal scheduling. However, in the rearrangement below, we will show that we can get no worse performance by presenting item $j$ for time $\delta$. Thus, an optimal ordering would never have $\delta < 0$ and we will henceforth assume $\delta \geq 0$.

We consider swapping the ordering of $i$ and $j$, presenting $i$ first for $\sigma_j + \gamma$ and then presenting $j$ for $\delta$. The clickthrough rate is unchanged beyond $\sigma_j + \gamma$, so we need only worry about the region before that. The optimal ordering achieved

$$\int_0^\gamma a_xg(x + \sigma_j) + \int_{\gamma}^{\gamma+\sigma_j} a_xg(x - \gamma) \, dx.$$  

When we perform the change mentioned above, the value in this region becomes

$$\int_0^{\gamma+\sigma_j} a_xg(x) \, dx.$$
The value we lose in the region from $\gamma$ to $\gamma + \sigma_j$ is $\int_\gamma^{\gamma + \sigma_j} a_x(g(x - \gamma) - g(x)) \, dx$.
However, this is offset by the gain in the region from 0 to $\gamma$ of $\int_0^{\gamma} a_x(g(x) - g(x + \sigma_j)) \, dx$. Again, the proof breaks down into two cases.

\textit{Case 1: $\sigma_j < \gamma$.}

Here the loss can be rewritten as

$$\int_0^{\sigma_j} a_x(g(x - \gamma) - g(x)) \, dx \leq a_\gamma \int_0^{\sigma_j} (g(x) - g(x + \gamma)) \, dx.$$  

The gain can be rewritten as

$$\int_0^{\gamma} a_x(g(x) - g(x + \sigma_j)) \, dx \geq a_\gamma \int_0^{\gamma} (g(x) - g(x + \gamma)) \, dx.$$  

Thus, the gain always exceeds the loss.

\textit{Case 2: $\sigma_j \geq \gamma$.}

Here the loss can be rewritten as

$$\int_0^{\sigma_j} a_x(g(x - \gamma) - g(x)) \, dx \leq a_\gamma \int_0^{\sigma_j} (g(x) - g(x + \gamma)) \, dx.$$  

The gain can be rewritten as

$$\int_0^{\gamma} a_x(g(x) - g(x + \sigma_j)) \, dx \geq a_\gamma \int_0^{\gamma} (g(x) - g(x + \sigma_j)) \, dx.$$  

Again, the gain always exceeds the loss.

Thus, in the case where the function $g$ exists and traffic is monotone, we can easily determine the ordering. Once the ordering is fixed (and using discrete time
intervals) we can use dynamic programming to solve the scheduling problem. If the total time is $T$ and there are $N$ items to schedule, we find the optimal scheduling for the first $t \leq T$ time and the first $n \leq N$ items, when they are ordered as in Lemma 6.4.3 and Lemma 6.4.4. If this optimal schedule has value $\text{opt}(t, n)$ we find it by finding $t'$ such that $\text{opt}(t', n - 1) + \text{value}(t', t, n)$ is maximized.

It is important to note that we do not need to actually know the values of $g(\cdot)$ and $\sigma$ for the algorithm to work. As long as they do exist and we can order the items according to $\sigma$, the actual values are unimportant to the running of the algorithm.

In the unimodal case, things are a bit more complicated. If we knew how to straddle the peak, and which side of the peak to put the items on, we could then schedule the increasing and decreasing parts independently. Of course there are exponentially many ways to divide the items into two groups, so a naive approach won’t work. However, with a more careful approach, we show

**Theorem 6.4.5.** When traffic is unimodal, we can find the optimal scheduling in polynomial time.

**Proof.** First note that Lemma 6.4.3 and Lemma 6.4.4 tell us nothing about the item that straddles the peak, but they show that the items on either side are ordered by $\sigma$ with smaller values closer to the peak. Thus, the item with largest $\sigma$, which we will denote item $n$, is either the first item in the entire sequence, or the last. This suggests a dynamic programming recurrence for computing the optimal ordering. We will denote the optimal value for the interval $[a, b)$ using the first $n$ items by $\text{opt}(a, b, n)$. There are two cases now. If item $n$ comes last, then $\text{opt}(a, b, n) = \text{opt}(a, t, n - 1) + \text{value}(t, b, n)$, for some $a \leq t \leq b$, while if item $n$ come first $\text{opt}(a, b, n) = \text{opt}(t, b, n - 1) + \text{value}(a, t, n)$, for some $a \leq t \leq b$. 
This recurrence omits the base case where an item straddles the peak. However, we can run the algorithm \(N\) times, trying each item for each straddling interval as the base case, and removing that item from the set to be scheduled to the sides. Each of the \(N\) times this takes time \(O(NT^3)\), so trying all \(N\) takes \(O(N^2T^3)\).

**Extending the algorithm to bimodal and K-modal cases**  This algorithm can be generalized to the case where we have two peaks instead of one. In this case, we must first start by picking the items that straddle the two maxima and the minima between them. Once this is done we can compute the optimal in a similar manner. We find that \(\text{opt}(t_1, t_2, t_3, t_4, n)\) is the optimal value when the first \(n\) items (where three have already been removed for minima and maxima) have been assigned and the intervals \([t_1, t_2]\) and \([t_3, t_4]\) are covered. Using a recurrence analogous to that for the unimodal piece, we try all four possible locations for the \(n\)-th item: to the left of the first peak, to the right of the first peak, to the left of the second peak, to the right of the second peak. Finally, we search over all \(\text{opt}(0, t_2, t_3, T, N - 3)\) and for each one we fill in the interval \([t_2, t_3]\) with the item we held out for the minima, picking the schedule with the highest value.

There are \(O(N^3)\) ways to pick the items for the minima and maxima, and once we have picked them, we have to find the value of \(\text{opt}(\cdot)\) for \(O(NT^4)\) configurations, which takes time \(O(NT^5)\). Putting this together, we get \(O(N^4T^5)\).

We can further extend this algorithm to the multimodal case, first fixing the items that straddle the peaks and valleys, and then running a dynamic programming algorithm analogous to that for two peaks. If there are \(K\) peaks, there are \(2K - 1\) peaks and valleys, and so we need to select out items to straddle these. Once those are selected, we must compute the optimal schedule for each set of in-
intervals \([a_1, b_1], [a_2, b_2], \ldots, [a_K, b_K]\), where the intervals do not overlap and \([a_i, b_i]\) straddles peak \(i\). In the worst case, this gives us a runtime of \(O(T^{2K+1}N^{2K}K^{-2})\).

### 6.4.3 Performance when conditions not met

Even when the exact conditions of unimodality and \(f_i(x) = g(x + \sigma_i)\) are not precisely met, the dynamic programming algorithm still gives a valid solution to the scheduling problem. In fact, if we find an ordering based on an approximation of \(\sigma\) and that ordering is close enough that it matches the optimal ordering, the dynamic programming algorithm will still choose the correct interval lengths. But, in the case where the ordering is wrong, we can still bound the error in our result by the degree to which the actual data deviates from our assumptions.

Given the actual data, we can find a unimodal traffic function such that the actual traffic is higher than the unimodal function, but exceeds it by at most a factor of \(\alpha\). Similarly, we can find a function \(g\) and values of \(\sigma_i\) such that \(g(x + \sigma_i) \leq f_i(x) \leq \beta g(x + \sigma_i)\). Any scheduling for the original problem gives no more clicks than a factor of \(\alpha\beta\) times the version meeting our conditions. Since we solve that version optimally, our algorithm gives a schedule that is within a factor of \(\alpha\beta\) optimal.

Figure 6.2 shows that the unimodality constraint is roughly met in our data, while Section 6.3 showed that all clickthrough rate functions could be aligned fairly well. In the next section we will see how close our algorithm comes to the optimal ordering on this real data.
Table 6.1: The percent improvement of our algorithm over the actual data, and over the simpler baseline algorithm. On all days, our algorithm is within 0.1% of optimal.

<table>
<thead>
<tr>
<th>Day</th>
<th>Percent Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Over Actual</td>
</tr>
<tr>
<td>1</td>
<td>26.0</td>
</tr>
<tr>
<td>2</td>
<td>12.1</td>
</tr>
<tr>
<td>3</td>
<td>66.9</td>
</tr>
<tr>
<td>4</td>
<td>28.2</td>
</tr>
<tr>
<td>5</td>
<td>18.5</td>
</tr>
<tr>
<td>6</td>
<td>21.4</td>
</tr>
<tr>
<td>7</td>
<td>36.3</td>
</tr>
<tr>
<td>8</td>
<td>28.9</td>
</tr>
<tr>
<td>9</td>
<td>20.4</td>
</tr>
<tr>
<td>10</td>
<td>34.6</td>
</tr>
<tr>
<td>11</td>
<td>27.1</td>
</tr>
<tr>
<td>12</td>
<td>18.0</td>
</tr>
<tr>
<td>13</td>
<td>22.2</td>
</tr>
<tr>
<td>14</td>
<td>34.3</td>
</tr>
<tr>
<td>15</td>
<td>18.2</td>
</tr>
<tr>
<td>16</td>
<td>23.6</td>
</tr>
<tr>
<td>17</td>
<td>18.9</td>
</tr>
<tr>
<td>18</td>
<td>25.0</td>
</tr>
<tr>
<td>19</td>
<td>18.7</td>
</tr>
<tr>
<td>20</td>
<td>20.1</td>
</tr>
<tr>
<td>21</td>
<td>23.5</td>
</tr>
<tr>
<td>Avg</td>
<td>25.9</td>
</tr>
</tbody>
</table>

6.5 Experiments

We start with the raw clickthrough rates for each of the articles in our dataset. For each article, we approximate the clickthrough rate function after presenting the article for $t$ minutes as $f_i(t) = e^{-\lambda(t+\sigma_i)}$, where $\lambda$ is a global decay parameter common to all articles and $\sigma_i$ measures each article’s inherent popularity as it is offset in time. To estimate $\lambda$, we picked the value that minimized the overall squared error (summed over all articles). In our dataset, we find $\lambda = 0.0044$, 


144
Figure 6.6: This figure shows our scheduling and the optimal scheduling for one particular day. Because the page views are not quite unimodal, our ordering is not quite optimal. It is close enough that the fraction lost is only 3.3E-5 though. For comparison, our baseline method does 2.2% worse, and we estimate a 26% improvement over the ordering that was used.

indicating that on average the clickthrough rate of an article declines by about 0.44% each minute.

To simulate our scheduling algorithm for a given day, we start by extracting the clickthrough rate data for each article actually used by Yahoo! that day. From this, we fit $f_i(t)$ by finding the best $\sigma_i$. We cannot simply use the true clickthrough rates because we only have that data for the time before the article was replaced on the site — using the fitted version allows us to simulate placing the articles for arbitrary amounts of time. In addition to the clickthrough rates, we extract data about the number of front-page views at each minute of the day. As our analysis in Section 6.3 shows, while an application would not have these exact numbers, they can be estimated very reliably from traffic in other weeks.

We know from Section 6.4 that our scheduling algorithm is optimal for this type of function, provided that the traffic is unimodal. The true traffic is close to
unimodal, but this condition is not strictly met. Thus, in evaluating our algorithm on this real data, we will consider three questions. First, how close is our algorithm to optimal, given that the optimality conditions are not strictly met? Second, how much better do we expect to do compared to the scheduling that was used? Third, how much better is our algorithm than a simple baseline? The baseline algorithm we compare to sets all the article lengths so that their final clickthrough rates (before being replaced) are about the same (as close as possible given the 1-minute resolution). It then orders them by placing the best article in the middle, straddling the peak, the next best to the right, the next to the left, and so on, alternating sides of the peak.

This problem is difficult enough that manual scheduling has fallen well short of optimality. First, our algorithm achieves 99.99% of the optimal algorithm, and is a 6.4% improvement over the simple baseline algorithm (with a daily maximum of a 13.7% improvement over the baseline). While this may seem modest, we should note that the number of daily visitors to the Yahoo! front page makes 6.5% a significant number. Finally, our algorithm gives a 25.9% improvement over the manual scheduling used at the time, a huge improvement.

One possible concern when examining these results is that some of the articles may not have been available to the human editors at Yahoo! until roughly when they were first scheduled. (Though, as discussed in the introduction to this chapter, these articles come from a class of applications where the content is not time-critical, so this is not necessarily a significant issue in reality.) To make sure that the improvements are not arising purely from the ability to shift content early, we run the optimal algorithm with one additional constraint: our algorithm may not schedule anything earlier than the first time it appeared on the Yahoo site. Even
with this constraint, we still do 17.6% better, indicating that our improvement is not coming simply because we have the ability to shift articles earlier in the day.

6.6 Generative models

Now that we have seen how the traffic data can be used to inform the design of our algorithms, it is interesting to ask whether we can explain the structure of this traffic data — particularly, the functional shape of the declining clickthrough rates — from more basic assumptions. In particular, the declining clickthrough rates are clearly arising from the aggregate behavior of the user population, who differ in their rates of activity on the Yahoo! front page and their response to the content that is presented there. A natural question is whether the clickthrough functions over time can be approximately derived from a few simple parameters quantifying
the variation within this user population. We show that this is indeed possible, suggesting some of the fundamental underlying mechanisms for the dependence of clickthrough rate on time.

The first step towards understanding this data is to look at how often users return to the front page and how likely they are to view a story once they get there. Each individual user has his or her own visit rate for the Yahoo! home page. Naturally, some users visit much more frequently than others and this plays an important role in scheduling because we want to serve fresh content to the frequent visitors. (This is one of the inherent trade-offs in the scheduling problem, as discussed earlier: we need to keep frequent visitors interested, but we have to avoid delivering low-quality content in the interest of freshness.) By examining server logs, we can easily determining the distribution of return rates.

Figure 6.7 shows the distribution of rates for all visitors to the Yahoo! home page over the course of one month. For the bulk of the distribution, we see that there is a good fit to a power law distribution with exponent $-1.5$.

A user whose overall rate is $\lambda$ will be expected to view the home page a number of times proportional to $\lambda$ over the course of a fixed time interval. However, the distribution of these views is another factor we must consider when modeling users. While it would be simplest to posit an exponential distribution for the interarrival times between home page views, the data shows that this is not accurate. Instead, Figure 6.8 shows that the time between views to the home page while a single article is featured is distributed as a power law with exponent roughly 0.5 and an exponential cutoff. If we condition on a user’s rate, the graph changes for different values of $\lambda$, but fitted exponential-cutoff power laws have similar powers, and mostly vary in the exponential cutoff term, which is highly correlated to $\lambda$. 
Figure 6.8: The distribution of interarrival gaps.

Figure 6.9: The clickthrough rate declines as visitors repeated see the same featured item.
Thus, we model a user with rate $\lambda$ as someone who returns repeatedly by sampling interarrival times from a distribution proportional to $x^{-0.5} \exp(-x\lambda)$ (which has mean close to $1/\lambda$).

The final step towards modeling users is to examine how the probability of a user clicking on an article depends on the number of times they have been exposed to it. Naturally, a user is most likely to click on an article the first time they are given the opportunity. Each time a user returns to the home page, the probability decreases as the user becomes attenuated to the article. Again, we go to the data to see how this attenuation works. Figure 6.9 shows the decreasing click through rate for one article as a visitor sees it over and over again: a power law with exponent $\gamma = -0.75$. We note that only a user’s first click on a featured article is considered when computing the clickthrough rate. Subsequent clicks are disregarded.

We now propose a model to combine the observed distributions. First, a user samples his or her overall rate $\lambda$ from the power law rate distribution. Once this rate is sampled, the user starts generating arrival gaps, starting at time $t_0 = 0$. To sample an arrival gap $\delta_i$, a user samples $\delta_i$ from a power law distribution with exponential cutoff, where the cutoff parameter corresponds to the rate, setting $t_i = t_{i-1} + \delta_i$. For some interval $[S,T]$, a featured article is presented on the home page. For each $S \leq t_i \leq T$ a user considers clicking on the article according to the attenuation function and the article’s inherent interest. Thus, if an article has inherent interest $K$, and $t_i$ is the $j$-th time the user has been to the home page in the interval $[S,T]$, the probability that the user will click is $K^j \gamma$. The final caveat is that a user who has clicked on an article will never click a second time.

Figure 6.10 shows the result of simulating this model for the parameters we see in the data. For comparison, the actual clickthrough rate for a specific article
Figure 6.10: Here we put together the visit rate distribution, interarrival gap distribution, and probability of clicking an article as a function of times seen on home page. The figure above shows the actual clickthrough rates along with simulated clickthrough rates for one article and a corresponding set of model parameters.

is also shown. While the curves differ somewhat, we see that the overall effect is quite similar. The fact that the simulated curve is produced purely from a very small number of aggregate parameters about the user population suggests that the functional shape of the declining clickthrough rates is largely rooted in simple collective properties of the set of users.

6.7 Discussion

We have formulated a general media scheduling problem, and shown how the types of data available at high-traffic websites makes it possible to develop algorithms that improve significantly on current practice in scheduling featured content. This problem is general enough that it can be applied to a wide variety of web sites. We have shown that real data is structured in a way that allows an efficient algorithm to schedule these items optimally.
There are a number of directions for further work. One interesting issue, noted earlier, is that while we have focused on applications where the pool of available content is known at the start of the scheduling period (e.g. at the start of the day), there are other applications — such as presenting breaking news — that are inherently on-line, with limited or zero advance information about content until it is ready to be scheduled. Related to this is content that may be available somewhat in advance, but whose utility function \( f_i(t) \) depends on when it is shown. We believe that the ideas developed in this chapter provide a useful starting point for thinking about these further variations on the problem, and this is an interesting direction for future work.

### 6.8 Related work

The related work falls into mainly three categories, namely, optimization and recommendation in online content-delivery systems, scheduling problems in the context of online user activities, and the dynamics of human response.

Das et al. [36] considered the problem of real-time recommendation of news articles to users. Their focus was on the recommendation and personalization aspects to maximize readership. Agarwal et al. [4] studied the problem of selecting and serving articles to users in an online portal to maximize the clickthrough rate. They develop a general framework to incorporate aspects such as learning, explore-exploit strategies, and individual user characteristics. While closest to theirs in end goal, our approach differs in being combinatorial while theirs is statistical. There is rich literature on improving website organization to optimize user experience; see, for instance, [114].
Szabo and Huberman [119] investigate methods for predicting the popularity of online content from user access patterns; they show long term popularity can be measured by the popularity at an early time. Using this, Wu and Huberman [132] study the role of popularity and novelty of an article and its position on a webpage in determining the growth of collective attention to the article; see also [133]. They formulate a model based on novelty and popularity and use it to find an optimal ordering of news stories to be displayed on a web page so as to maximize the number of clicks in a finite horizon. Their concern is more about the spatial display of news articles (modeling factors such as novelty) whereas we are interested in a temporal ordering of items.

Dasgupta et al. [37] considered a job scheduling problem that is motivated by web advertising. In this setting items (stories) arrive online, each with a length and per-unit value, and the goal is to pre-emptively schedule them to maximize the total time-discounted value. The main difference between their work and ours is offline vs online: our goal is to construct an offline schedule of items whereas their goal is to obtain an online schedule that is competitive against an offline optimum. For a detailed description of many job scheduling algorithms, we refer to [25, 86].

Barabasi [12] argued that the bursty nature of human behavior is caused by a decision-based queuing process, where tasks executions are priority-driven, with the timing of execution heavy-tailed; for further work, see [121, 122]. Johansen and Sornette [67] and subsequently Johansen [68] studied the response of online users to a “point-like” perturbation (e.g., publication of an online article) and showed that the download rate of the article is inverse polynomial. A similar observation was made between an email message and its reply [69]. Oliveira and Barabasi [105] show that the correspondence patterns of Darwin and Einstein is similar to
today’s electronic correspondences. Our analysis of user return times and interest levels provides a further perspective on how response times vary across a very large population in different applications, in our case for the purpose of informing an underlying scheduling algorithm.
CHAPTER 7
ANONYMIZED SOCIAL NETWORKS AND STRUCTURAL STEGANOGRAPHY

As we saw in the preceding chapters, digital traces of human social interactions can now be found in a wide variety of on-line settings, and this has made them rich sources of data for large-scale studies of social networks. While a number of these on-line data sources are based on publicly crawlable blogging and social networking sites [10, 80, 89], where users have explicitly chosen to publish their links to others, many of the most promising opportunities for the study of social networks are emerging from data on domains where users have strong expectations of privacy — these include e-mail and messaging networks, as well as the link structure of closed (i.e. “members-only”) on-line communities [1, 2, 60, 75, 82]. As a useful working example, consider a “communication graph,” in which nodes are e-mail addresses, and there is a directed edge \((u, v)\) if \(u\) has sent at least a certain number of e-mail messages or instant messages to \(v\), or if \(v\) is included in \(u\)’s address book. Here we will be considering the “purest” form of social network data, in which there are simply nodes corresponding to individuals and edges indicating social interaction, without any further annotation such as time-stamps or textual data.

In designing studies of such systems, one needs to set up the data to protect the privacy of individual users while preserving the global network properties. This is typically done through anonymization, a simple procedure in which each individual’s “name” – e.g., e-mail address, phone number, or actual name – is replaced by a random user ID, but the connections between the (now anonymized) people – encoding who spoke together on the phone, who corresponded with whom, or who instant-messaged whom – are revealed. The motivation behind anonymizing
is roughly as follows: while the social network labeled with actual names is sensitive and cannot be released, there may be considerable value in allowing researchers to study its structure. For such studies, including those cited above, researchers are not specifically interested in “who” corresponds to each node, but in the properties of the graph, such as its connectivity, node-to-node distances, frequencies of small subgraphs, or the extent to which it can be clustered. Anonymization is thus intended to exactly preserve the pure unannotated structure of the graph while suppressing the “who” information.

Can this work? The hope is that being handed an anonymized picture of a social network — just a graph with a random identifier attached to each node — is roughly akin to being given the complete social network of Mars, with the true Martian names attached to the nodes. Intuitively, the names are meaningless to earth-dwellers: we do not “know” the Martians, and it is completely irrelevant to us whether a given node in the graph is labeled “Groark” or “Zoark”. The difficulty with this metaphor, of course, is that anonymous social network data almost never exists in the absence of outside context, and an adversary can potentially combine this knowledge with the observed structure to begin compromising privacy, de-anonymizing nodes and even learning the edge relations between explicitly named (de-anonymized) individuals in the system. Moreover, such an adversary may in fact be a user (or set of users) of the system that is being anonymized.

For distinguishing among ways in which an adversary might take advantage of context, it is useful to consider an analogy to the distinction between passive attacks and active attacks in cryptanalysis — that is, between attacks in which an adversary simply observes data as it is presented, and those in which the adversary actively tries to affect the data to make it easier to decipher. In the case
of anonymized social networks, passive attacks are carried out by individuals who try to learn the identities of nodes only after the anonymized network has been released. In contrast, an adversary in an active attack tries to compromise privacy by strategically creating new user accounts and links before the anonymized network is released, so that these new nodes and edges will then be present in the anonymized network.

### 7.1 Attacks on anonymized social networks

In this chapter we present both active and passive attacks on anonymized social networks, showing that both types of attacks can be used to reveal the true identities of targeted users, even from just a single anonymized copy of the network, and with a surprisingly small investment of effort by the attacker.

We describe active attacks in which an adversary chooses an arbitrary set of users whose privacy it wishes to violate, creates a small number of new user accounts with edges to these targeted users, and creates a pattern of links among the new accounts with the goal of making it stand out in the anonymized graph structure. The adversary then efficiently finds these new accounts together with the targeted users in the anonymized network that is released. At a theoretical level, the creation of $O(\sqrt{\log n})$ nodes by the attacker in an $n$-node network can begin compromising the privacy of arbitrary targeted nodes, with high probability for any network; in experiments, we find that on a 4.4-million-node social network, the creation of 7 nodes by an attacker (with degrees comparable to those of typical nodes in the network) can compromise the privacy of roughly 2400 edge relations on average. Moreover, experimental evidence suggests that it may be very difficult to determine whether a social network has been compromised by such an active attack.
We also consider passive attacks, in which users of the system do not create any new nodes or edges — they simply try to find themselves in the released network, and from this to discover the existence of edges among users to whom they are linked. In the same 4.4-million-node social network dataset, we find that for the vast majority of users, it is possible for them to exchange structural information with a small coalition of their friends, and subsequently uniquely identify the subgraph on this coalition in the ambient network. Using this, the coalition can then compromise the privacy of edges among pairs of neighboring nodes.

There are some obvious trade-offs between the active and passive attacks. The active attacks have more potent effects, in that they are guaranteed to work with high probability in any network (they don’t force users to rely on the chance that they can uniquely find themselves after the network is released), and the attacker can choose any users it wants to target. On the other hand, while the passive attack can only compromise the privacy of users linked to the attacker, it has the striking feature that this attacker can simply be a user of the system who sees the anonymized network and indulges his or her curiosity; there is no observable “wrongdoing” to be detected. Moreover, since we find in practice that the passive attack will succeed for the majority of the population, it says in effect that most people in a large social network have laid the groundwork for a privacy-breaching attack simply through their everyday actions, without even realizing it.

These trade-offs naturally suggest the design of hybrid “semi-passive” attacks, in which a user of the system creates no new accounts, but simply creates a few additional out-links to targeted users before the anonymized network is released. As we show later, this can lead to privacy breaches on a scale approaching that of
the active attack, without requiring the creation of new nodes.

We now summarize the results more fully, before moving on in subsequent sections to the details behind them.

7.1.1 The nature of the attacks

We assume the social network is an n-node graph $G = (V,E)$, representing interactions in an on-line system. Nodes correspond to user accounts, and an edge $(u,v)$ indicates that $u$ has communicated with $v$ (again, consider the example of an e-mail or instant messaging network). The attacks become easier to carry out if the released graph data is directed; for most of the paper we will therefore consider the harder case of undirected graphs, in which we assume that the curator of the data — the agent that releases the anonymized network — eliminates the directions on the edges.

The active attacks will make use of the following two types of operations. First, an individual can create a new user account on the system; this adds a new node to $G$. Second, a node $u$ can decide to communicate with a node $v$; this adds the undirected edge $(u,v)$ to $G$. The goal of the attack is to take an arbitrary set of targeted users $w_1, \ldots, w_b$, and for each pair of them, to use the anonymized copy of $G$ to learn whether the edge $(w_i, w_j)$ in fact exists. This is the sense in which the privacy of these users will be compromised. (Other privacy compromises, such as learning the degree of a targeted user, also occur, but we focus our attention on learning about edges.)

The structure of the active attack is roughly as follows. Before the anonymized graph is produced, the attacker creates $k$ new user accounts (for a small parameter
and it links them together to create a subgraph $H$. It then uses these accounts to create links (e.g. by sending messages or creating address book entries) to nodes in $\{w_1, \ldots, w_b\}$, and potentially other nodes as well. Now, when the anonymized copy of $G$ is released, this subgraph $H$ will be present, as will the edges connecting $H$ to $w_1, \ldots, w_b$. The attacker finds the copy of $H$ that it planted in $G$, and from this it locates $w_1, \ldots, w_b$. Having identified the true location of these targeted users in $G$, the attacker can then determine all the edges among them, thereby compromising privacy.

There are a number of challenges in making this high-level approach actually work. First, if only a single copy of $G$ is going to be released, then the attacker needs to construct $H$ before having seen the structure of $G$. This means constructing a subgraph $H$ that is likely to be uniquely identifiable in $G$, regardless of what $G$ looks like. Second, the attacker needs to be able to efficiently find its copy of $H$ hidden within $G$ — in other words, it needs to create an instance of the subgraph isomorphism problem that is tractable to solve, even in a graph $G$ with several million nodes.

The passive attack is based on the observation that most nodes in real social network data already belong to a small uniquely identifiable subgraph. Hence, if a user $u$ is able to collude with a coalition of $k - 1$ friends after the release of the network, he or she will be able to identify additional nodes that are connected to this coalition, and thereby learn the edge relations among them.

It is also worth noting, however, that even the active attacks only involve the use of completely innocuous operations in the context of the system being compromised — the creation of new accounts, and the creation of links to existing accounts. In this sense, while the active attacker’s aims are nefarious (and, in
almost any imaginable scenario, prohibited either by research ethics guidelines or the terms of service of the system, or both), none of the individual steps from which the attack is constructed could be viewed at a syntactic level as “breaking into” parts of the system where it is not allowed. We also note, without going into the technical details here, that the active attacks are not something that degrade if many people are trying to execute them: even if many separate parties simultaneously run copies of the active attack, the high probability outcome is that all of them will succeed.

### 7.1.2 Parameters of the active attacks

For an active attacker to produce a subgraph $H$ likely to be unique in the network, it can use random generation: the attacker creates $k$ new user accounts, and produces links among them by creating an edge between each pair independently at random.

We present two different active attacks employing this high-level idea, but differing in their specifics. For the first attack, we show that with $k = \Theta(\log n)$ new accounts, a randomly generated subgraph $H$ will be unique with high probability, regardless of what $G$ looks like and regardless of how $H$ is attached to the rest of $G$. Moreover, if the maximum node degree in $H$ is $\Theta(\log n)$, then $H$ is recoverable efficiently, together with the identities of up to $b = \Theta(\log^2 n)$ targeted nodes to whom the attacker created links from $H$. The recovery algorithm for $H$ uses a search over short walks in $G$, and accordingly we call it the *walk-based attack*.

In practice, $k$ can be set to values even smaller than the bounds suggest, and recovery is very efficient. In computational experiments on a 4.4-million-node
social network, a subgraph built using \( k = 7 \) new nodes, and degrees comparable
to those of typical nodes in the network, can reveal an average of 70 targeted nodes,
and hence the \( \binom{70}{2} = 2415 \) edge relations among them. We also provide evidence
that it may be hard to detect whether such a subgraph \( H \) has been inserted into
\( G \); we will discuss the issue of detection in more detail below. Finally, we note
that for the passive attack, we use the efficient recovery algorithm designed for
this walk-based attack in order to identify a small coalition of existing nodes in
the anonymized network.

The second active attack is similar in flavor; it also constructs \( H \) by including
edges at random, but it attaches \( H \) to \( G \) using very few edges and recovers it using
a more complex computation based on Gomory-Hu cut trees [54, 61]. Hence we
will refer to it as the cut-based attack. The “thin” attachment of \( H \) to the rest of
\( G \) implies that \( H \) will likely be unique and efficiently findable at an asymptotically
smaller value of \( k \): the cut-based attack uses \( k = O(\sqrt{\log n}) \) to reveal the identities
of \( \Theta(\sqrt{\log n}) \) targeted nodes.

There are some trade-offs between the two active attacks. The walk-based at-
tack comes with an extremely fast recovery algorithm that easily scales to millions
of nodes, and it appears to be very hard to detect. The cut-based attack has
the advantage of matching the tight theoretical bound on the number of nodes
needed — we can show that an attacker must create at least \( \Omega(\sqrt{\log n}) \) new nodes
in the worst case to begin compromising the privacy of arbitrary targeted nodes.
The use of Gomory-Hu trees in the cut-based attack makes its recovery algorithm
more expensive than that of the walk-based attack (though see the recent successes
with Gomory-Hu computations on large-scale network analysis in [54]). Finally,
the walk-based attack has the potential to compromise \( \Theta(k^2) \) users, while the cut-
based attack can only compromise $O(k)$, and it also appears easier to detect that the cut-based attack has taken place.

### 7.2 The Walk-Based Attack

We begin by describing the specifics of the walk-based attack; we then analyze the method in Section 7.2.2, and report on computational experiments with it in Section 7.2.3.

#### 7.2.1 Description of the Attack

Let $G = (V, E)$ be the $n$-node graph representing the anonymized social network that is released. As noted above, we consider the undirected case, in which there is an undirected edge $(u, v)$ if at least one of the directed edges $(u, v)$ or $(v, u)$ is present. We focus on the undirected case because the attack becomes easier if the graph is directed.

Let us consider the problem from the perspective of the attacker. For ease of presentation, we begin with a slightly simplified version of the attack, and then show how to extend it to the attack we really use. We first choose a set of $k$ named users, $W = \{w_1, \ldots, w_k\}$, that we wish to target in the network — we want to learn all the pairs $(w_i, w_j)$ for which there are edges in $G$. To find each $w_i$ in the anonymized graph, we use the following strategy. We first create a set of $k$ new user accounts, $X = \{x_1, \ldots, x_k\}$, which will appear as nodes in the system. We include each undirected edge $(x_i, x_j)$ independently with probability $1/2$. This produces a random graph $H$ on $X$. 

163
We also create an edge \((x_i, w_i)\) for each \(i\). (As discussed above, this involves having \(x_i\) send \(w_i\) a message, or include \(w_i\) in an address book, or some other activity depending on the nature of the social network.) For describing the basic version of the attack, we also assume that, because the account \(x_i\) corresponds to a fake identity, it will not receive messages from any node in \(G - H\) other than potentially \(w_i\), and thus will have no link to any other node in \(G - H\). However, we will see later that the attack can be made to work even when this latter assumption does not hold.

When the anonymized graph \(G\) is released, we need to find our copy of \(H\), and to correctly label its nodes as \(x_1, \ldots, x_k\). Having found these nodes, we then find \(w_i\) as the unique node in \(G - H\) that is linked to \(x_i\). We thus identify the full labeled set \(W\) in \(G\), and we can simply read off the edges between its elements by consulting \(G\).

A number of technical ingredients are needed in order to make this plan work, based on whether certain subgraphs have the same structure as each other, and whether they have any internal symmetries. To express such questions, we use the following terminology. For a set of nodes \(S\), we let \(G[S]\) denote the subgraph of \(G\) induced by the nodes in \(S\). An isomorphism between two sets of nodes \(S\) and \(S'\) in \(G\) is a one-to-one correspondence \(f : S \rightarrow S'\) that maps edges to edges and non-edges to non-edges: \((u, v)\) is an edge of \(G[S]\) if and only if \((f(u), f(v))\) is an edge of \(G[S']\). In this case, \(G[S]\) and \(G[S']\) are isomorphic — they are the same graph up to relabeling. An automorphism is an isomorphism from a set \(S\) to itself — a relabeling of the nodes \(f : S \rightarrow S\) that preserves graph’s structure. An automorphism \(f\) is non-trivial if it is not the identity function.

Thus, the construction of \(H\) succeeds if

164
(i) there is no $S \neq X$ such that $G[S]$ and $G[X] = H$ are isomorphic;
(ii) the subgraph $H$ can be efficiently found, given $G$; and
(iii) the subgraph $H$ has no non-trivial automorphisms.

If (i) holds, then any copy of $H$ we find in $G$ must in fact be the one we constructed;
if (ii) holds, then we can in fact find the copy of $H$ quickly; and if (iii) holds, then
once we find $H$, we can correctly label its nodes as $x_1, \ldots, x_k$, and hence find
$w_1, \ldots, w_k$.

The full construction is almost as described above, with the following three
additions. First, the size of the targeted set $W$ can be larger than $k$. The idea is
that rather than connect each $w_i$ to just a single $x_i$, we can connect it to a subset
$N_i \subseteq X$, as long as $w_i$ is the only node in $G - H$ that is attached to precisely
the nodes in $N_i$ — this way $w_i$ will still be uniquely identifiable once $H$ is found.
Second, we will explicitly randomize the number of links from each $x_i$ to $G - H$,
to help in finding $H$. And third, to recover $H$, it is helpful to be able to traverse
its nodes in order $x_1, x_2, \ldots, x_k$. Thus, we deterministically include all edges of the
form $(x_i, x_{i+1})$, and randomly construct all other edges.

The Construction of $H$ With this informal discussion in mind, we now give
the full specification of the attack.

(1) We choose $k = (2 + \delta) \log n$, for a small constant $\delta > 0$, to be the size of
$X$. We choose two constants $d_0 \leq d_1 = O(\log n)$, and for each $i = 1, 2, \ldots, k$, we
choose an external degree $\Delta_i \in [d_0, d_1]$ specifying the number of edges $x_i$ will have
to nodes in $G - H$. Each $\Delta_i$ can be chosen arbitrarily, but in our experiments with
the algorithm, it works well simply to choose each $\Delta_i$ independently and uniformly
at random from the interval $[d_0, d_1]$. 165
(2) Let \( W = \{w_1, w_2, \ldots, w_b\} \) be the users we wish to target, for a value \( b = O(\log^2 n) \). We also choose a small integer constant \( c \) \((c = 3 \ \text{will suffice in what follows})\). For each targeted node \( w_j \), we choose a set \( N_j \subseteq \{x_1, \ldots, x_k\} \) such that all \( N_j \) are distinct, each \( N_j \) has size at most \( c \), and each \( x_i \) appears in at most \( \Delta_i \) of the sets \( N_j \). (This gives the true constraint on how large \( b = O(\log^2 n) \) can be.)

We construct links to \( w_j \) from each \( x_i \in N_j \).

(3) Before generating the random internal edges of \( H \), we add arbitrary further edges from \( H \) to \( G \setminus H \), so that each node \( x_i \) has exactly \( \Delta_i \) edges to \( G \setminus H \). We construct these edges subject only to the following condition: for each \( j = 1, 2, \ldots, b \), there should be no node in \( G \setminus H \) other than \( w_j \) that is connected to precisely the nodes in \( N_i \).

(4) Finally, we generate the edges inside \( H \). We include each edge \((x_i, x_{i+1})\), for \( i = 1, \ldots, k-1 \), and we include each other edge \((x_i, x_j)\) independently with probability \( 1/2 \). Let \( \Delta_i' \) be the degree of \( x_i \) in the full graph \( G \) (this is \( \Delta_i \) plus its number of edges to other nodes in \( X \)).

This concludes the construction. As a first fact, we note that standard results in random graph theory (see e.g. [17]) imply that with high probability, the graph \( H \) has no non-trivial automorphisms. We will assume henceforth that this event occurs, i.e., that \( H \) has no non-trivial automorphisms.

**Efficiently recovering \( H \) given \( G \)**  
When the graph \( G \) is released, we identify \( H \) by performing a search, with pruning, to find the path \( x_1, x_2, \ldots, x_k \). We start at every node in \( G \) of degree \( \Delta_i' \), and we successively try to add any node that has the correct degree and then correct edges back to nodes already on the path. In this way, we are pruning the search based on two kinds of tests: a degree test,
that each possible candidate for node $x_i$ should have the correct degree $\Delta_i'$; and an
internal structure test, that each possible candidate for node $x_i$ should have edges
to the correct subset of $\{x_1, x_2, \ldots, x_{i-1}\}$.

Here is full description of the search algorithm.

(1) A rooted search tree $T$ represents the progress of our search. Each node $\alpha$
in $T$ other than the root corresponds to a node in $G$, which we will denote $f(\alpha)$,
and the same node in $G$ can potentially appear multiple times in $T$. We construct
$T$ so that for every path of nodes $\alpha_1, \ldots, \alpha_\ell$ from the root, the corresponding
nodes $f(\alpha_1), \ldots, f(\alpha_\ell)$ form a path in $G$ with the same degree sequence and same
internal edges as $x_1, \ldots, x_\ell$; and conversely every such path in $G$ corresponds to a
distinct rooted path in $T$.

(2) We construct $T$ by initially creating a dummy root node $\alpha^*$. At any
intermediate point in the construction, we take each current leaf node $\alpha$, with a
path $\alpha^* = \alpha_0, \alpha_1, \ldots, \alpha_\ell = \alpha$ leading to it, and we find all neighbors $v$ of $f(\alpha)$ in $G$
for which the degree of $v$ is $\Delta_{\ell+1}'$, and $(f(\alpha_i), v)$ is an edge if and only if $(x_i, x_{\ell+1})$
is an edge for each $i = 1, \ldots, \ell$. For each such $v$, we create a new child $\beta$ of $\alpha$,
with $f(\beta) = v$.

(3) Finally, if there is a unique rooted length-$k$ path in $T$, then this must
correspond to the nodes of $H$, in order. Having found $H$, we then find the targeted
nodes $w_1, \ldots, w_b$ owing to the fact that $w_j$ is the only node with connections to
precisely the nodes in $N_j$. Note that the total running time is only a small factor
larger than the size of $T$. 

167
7.2.2 Analysis

To prove the correctness and efficiency of the attack, we show two things: with high probability the construction produces a unique copy of $H$ in $G$, and with high probability, the search tree $T$ in the recovery algorithm does not grow too large. It is important to stress that although these proofs are somewhat intricate, this complexity is an aspect of the analysis, not of the algorithms themselves. The construction of $H$ and the recovery algorithm have already been fully specified in the previous subsection, and they are quite simple to implement. In keeping with this, we have structured this subsection and the next (on computational experiments) so they can be read essentially independently of each other.

We begin with the uniqueness result.

**Theorem 7.2.1.** Let $k \geq (2 + \delta) \log n$ for an arbitrary positive constant $\delta > 0$, and suppose we use the following process to construct an $n$-node graph $G$:

(i) We start with an arbitrary graph $G'$ on $n - k$ nodes, and we attach new nodes $X = \{x_1, \ldots, x_k\}$ arbitrarily to nodes in $G'$.

(ii) We build a random subgraph $H$ on $X$ by including each edge $(x_i, x_{i+1})$ for $i = 1, \ldots, k - 1$, and including each other edge $(x_i, x_j)$ independently with probability $1/2$.

Then with high probability there is no subset of nodes $S \neq X$ in $G$ such that $G[S]$ is isomorphic to $H = G[X]$.

**Proof.** To begin, let $\mathcal{F}_0$ be the event that there is no subset of nodes $S$ disjoint from $X$ such that $G[S]$ is isomorphic to $H$. (Note the difference between $\mathcal{F}_0$ and the statement of the theorem — in $\mathcal{F}_0$, we require that $S$ be disjoint from $X$, not
just unequal to $X$.) We first prove

**Claim 1.** *With high probability, the event $\mathcal{F}_0$ holds.*

We can prove Claim 1 by adapting a short argument with its roots in lower bounds for Ramsey numbers [3, 50]. For an ordered sequence $S = (s_1, s_2, \ldots, s_k)$ of $k$ nodes in $G - H$, let $\mathcal{E}_S$ denote the event that the function $f : S \to X$ given by $f(s_i) = x_i$ is an isomorphism. Since all but $k - 1$ of the edges in $H$ are chosen independently with probability $1/2$, and since $S$ is disjoint from $X$, we have

$$\Pr[\mathcal{E}_S] = 2^{-\binom{k}{2} + (k-1)} = 2^{-\binom{k-1}{2}} = 2^{\frac{k+3}{2} - \frac{k^2}{2}}.$$

Now $\mathcal{F}_0 = \bigcup_S \mathcal{E}_S$, where the union is over all sequences of $k$ nodes from $G - H$. There are fewer than $n^k$ such sequences $S$, so using the Union Bound and the fact that $n \leq 2^{k/(2+\delta)}$,

$$\Pr[\mathcal{E}] < n^k \cdot 2^{-\binom{k-1}{2}} \leq 2^{k^2/(2+\delta)} \cdot 2^{1+3k/2} \cdot 2^{-k^2/2} = 2^{[-\delta k^2/2(2+\delta)] + 3k/2 + 1},$$

which goes to 0 exponentially quickly in $k$. This completes the proof of Claim 1.

This short argument for why $\mathcal{F}_0$ holds provides the technical intuition behind the more general statement of the theorem. All the remaining complexity in the proof comes from sets $S$ that may partially overlap $X$ — and indeed this is the trickier kind of $S$ to deal with, since one can try to construct an isomorphism with $H$ by combining large parts of $H$ with a few extra nodes elsewhere in the graph.

Due to this complexity, we need two facts asserting that $H$ does not have much internal symmetry. For the second, we use the following definition: a node $v$ is a *fixed point* of an isomorphism $f : S \to S'$ if $v \in S \cap S'$ and $f(s) = s$.

**Claim 2.** *For any constant $c_1 > 4$, let $\mathcal{F}_1$ denote the event that there are no disjoint sets of nodes $Y$ and $Z$ in $H$, each of size $c_1 \log k$, such that $H[Y]$ and*
are isomorphic. With high probability, the event $F_1$ holds.

(Claim 3.) Suppose event $F_1$ holds; then for any constant $c_2 \geq 3c_1$ the following holds. Let $A$, $B$, and $Y$ be disjoint sets of nodes in $G$, with $B, Y \subseteq X$, and let $f : A \cup Y \to B \cup Y$ be an isomorphism. Then the set $f(A)$ contains at most $c_1 \log k$ nodes not in $B$, and the set $Y$ contains at most $c_2 \log k$ nodes that are not fixed points of $f$.

The proof of Claim 2 closely parallels that of Claim 1, and we omit this proof here.

To prove Claim 3, we build the following directed graph $K$ on $A \cup B \cup Y$: if $f(v) = w$, then we include a directed edge from $v$ to $w$. Note that in $K$, nodes in $A$ have out-degree 1 and in-degree 0, nodes in $B$ have out-degree 0 and in-degree 1, and nodes in $Y$ have out-degree 1 and in-degree 1. Thus $K$ consists of node-disjoint paths, cycles, and self-loops, with the cycles and self-loops fully in $Y$, and each path beginning at a node in $A$, possibly passing through nodes in $Y$, and ending at a node in $B$. We say a path component of $K$ is non-trivial if it includes at least one node of $Y$.

First, note that there can be at most $c_1 \log k$ non-trivial path components in $K$; otherwise, if we let $Y' \subseteq Y$ consist of all the penultimate nodes on these paths, and $f(Y') = B' \subseteq B$, then $Y'$ and $B'$ are disjoint subsets of $X$, of size more than $c_1 \log k$ each, for which $H[Y']$ and $H[B']$ are isomorphic. This contradicts the assumption that $F_1$ holds. It follows that $f(A)$ contains at most $c_1 \log k$ nodes not in $B$.

Next, let $Z$ be the set of nodes in $Y$ that are not fixed points of $f$. Nodes in $Z$ correspond to the nodes on the cycle components in $K$, and the interior nodes on the path components. Suppose we choose every other edge on each cycle and each
path (starting with the second edge on each path): we obtain at least \( |Z|/3 \) edges, since the worst case is a length-3 cycle, where we get only one edge. Let \( Z_1 \subseteq Z \) be the tails of all these edges, and let \( Z_2 \subseteq Z \cup B \) be their heads. Then \( f(Z_1) = Z_2 \), and so \( G[Z_1] \) and \( G[Z_2] \) are isomorphic. But \( Z_1 \) and \( Z_2 \) are disjoint subsets of \( X \), so since \( \mathcal{F}_1 \) holds, we have \( |Z_1| = |Z_2| \leq c_1 \log k \), and hence \( |Z| \leq 3c_1 \log k \leq c_2 \log k \).

This completes the proof of Claim 3.

Finally, we set up the calculation that will conclude the proof of the theorem. Suppose events \( \mathcal{F}_0 \) and \( \mathcal{F}_1 \) hold, and that there is a non-empty set \( A \subseteq V - X \) such that, for some non-empty \( Y \subseteq X \), the subgraph \( G[A \cup Y] \) is isomorphic to \( H \). Let \( f : A \cup Y \to X \) be the isomorphism, and \( B = X - Y \). Let \( C = f(A) \), and \( D \) be the set consisting of all nodes of \( Y \) that are not fixed points of \( f \). By Claim 3, we have \( |C - B| \leq c_1 \log k \) and \( |D| \leq c_2 \log k \). Thus, if \( j = |A| = |B| = |C| \), then the set of fixed points \( Y' = Y - D - C \) has size at least \( k - (c_1 + c_2) \log k - j \). We write \( k' = k - (c_1 + c_2) \log k \); since \( k = (2 + \delta) \log n \), we have \( k' \geq (2 + 2\delta_1) \log n \) for a smaller constant \( \delta_1 > 0 \) and \( n \) sufficiently large.

To show that there is unlikely to be a second copy of \( H \) in \( G \), we search over all possible choices for \( A, B, C, \) and \( D \) within the appropriate size bounds. (We keep track of the order of the elements in \( A \) and \( C \), which encodes the bijection between them.) Thus, let \( \mathcal{E}_{ABCD} \) be the event that \( G[A \cup Y] \) is isomorphic to \( H \) (where \( Y = X - B \)), via an isomorphism \( f \) in which \( C = f(A) \) and all elements in \( Y' = Y - D - C \) are fixed points of \( f \). At most \( j - 1 \) edges inside \( C \) belong to the path \( x_1, x_2, \ldots, x_k \) for which edges were explicitly included; thus, at least \( \binom{j}{2} - (j - 1) \) edges inside \( C \) are randomly generated. In order for \( \mathcal{E}_{ABCD} \) to hold, all of these must match the corresponding edges inside \( A \) (recall that we are keeping track of the ordering of \( A \) and \( C \)). Similarly, the \( \geq (k' - j)j - 2j \) random edges

171
created between $C$ and $Y'$ match those between $A$ and $Y'$.

Since $(k' - j)j + \left(\frac{j}{2}\right) - 3j \geq \frac{1}{2}k'j - \frac{7}{2}j$, we have

$$\Pr[\mathcal{E}_{ABCD}] \leq 2^{-\frac{1}{2}k'j + \frac{7}{2}j} \leq 2^{-j(1+\delta_1)\log n}2^{\frac{7}{2}j} = n^{-(1+\delta_1)j}2^{\frac{7}{2}j}.$$  

Finally,

$$\Pr[\mathcal{E}] \leq \sum_{A, B, C, D} \Pr[\mathcal{E}_{ABCD}] \leq \sum_{j \geq 1} n^j k^{2j} k^{c_2 \log k} n^{-(1+\delta_1)j} 2^{\frac{7}{2}j} \leq \sum_{j \geq 1} k^{c_2 \log k} \left(\frac{2^{\frac{7}{2}k^2}}{n^{\delta_1}}\right)^j,$$

and this last expression goes to 0 as $n$ increases.  

Since the running time of the recovery algorithm is only a small factor larger than the total number of nodes in the search tree $T$, we can bound the running time by bounding the size of this tree.

**Theorem 7.2.2.** For every $\varepsilon > 0$, with high probability the size of $T$ is $O(n^{1+\varepsilon})$.

**Proof.** Recall that $k$ denotes the size of $H$, and let $d$ be the maximum degree (in $G$) of a node in $H$. Both of these quantities are $O(\log n)$. Let $\Gamma'$ be a random variable equal to the number of paths of $G - H$ corresponding to some node of $T$, and let $\Gamma''$ be the number of paths in $G$ that meet $H$ and correspond to some node in $T$. Then $\Gamma = \Gamma' + \Gamma''$ is the number of nodes in $T$, the quantity we are seeking to bound. We will show how to bound $E[\Gamma]$, assuming that the events $\mathcal{F}_0$ and $\mathcal{F}_1$ from the proof of Theorem 7.2.1 hold.

We first bound $E[\Gamma']$, as follows. For a path $P$ in $G - H$, let $\Gamma'_P = 1$ if $P$ corresponds to a node in $T$, and $\Gamma'_P = 0$ otherwise. Call $P$ feasible if the degree of every node on $P$ is at most $d$. If $P$ is not feasible, then $\Gamma'_P = 0$ with probability
1. Now consider a feasible $P$ of length $j \leq k$; for $P$ to be represented in $T$, we need the edges among nodes on $P$ to match the edges among $\{x_1, x_2, \ldots, x_j\}$. We can imagine the edges among $\{x_1, x_2, \ldots, x_j\}$ (other than those on the known path $x_1, x_2, \ldots, x_j$) as being generated after $P$ is chosen, so $E[\Gamma'_p] = \Pr[\Gamma'_p = 1] = 2^{-(j-1)}$. The total number of feasible $P$ of length $j$ is at most $nd^{j-1}$. Thus,

$$E[\Gamma'] \leq n \sum_{j=1}^{k} d^{j-1}2^{-(j-1)} = n \sum_{j=1}^{k} (d/2^{j-2})^{j-1}. $$

Once $j$ is $\Theta(\log \log n)$, each term inside the sum is $O(1)$, so

$$E[\Gamma'] \leq nk d^{O(\log \log n)} = O(n2^{O((\log \log n)^2)}).$$

Now we bound $\Gamma''$, by decomposing it into a separate random variable for each possible pattern in which a path could snake in and out of $H$. Thus, we say that a template $\tau$ is a sequence of $\ell \leq k$ symbols $(\tau_1, \ldots, \tau_\ell)$, where symbol $\tau_i$ is either a distinct node of $H$ or a special symbol *. We call the set of all nodes of $H$ that appear in $\tau$ the support of $\tau$, and denote it $s(\tau)$. We will say that a path $P$ in $G$ is associated with $\tau$ if the $i^{th}$ node on $P$ lies in $G - H$ for $\tau_i = *$, and otherwise is equal to $\tau_i \in X$. Finally, we say that the reduction of $\tau$, denoted $\tau$, is the template for which $\tau_i = *$ whenever $\tau_i = *$, and for which $\tau_i = x_i$ otherwise. (We will call such a $\tau$ a reduced template.)

Let $\Gamma''_{\tau}$ be a random variable equal to the number of paths $P$ associated with $\tau$ that are represented in the search tree $T$. If at least one such path exists, then there is an isomorphism $f : s(\tau) \rightarrow s(\tau')$ given by $f(x_r) = x_i$ when $\tau_i = x_r$. Since we are assuming that $F_1$ holds, Claims 2 and 3 from the proof of Theorem 7.2.1 imply that at all but at most $O(\log k)$ nodes are fixed points of $f$, and hence that $\tau$ agrees with $\tau'$ on all but $O(\log k)$ positions. Hence, the only templates $\tau$ for which $\Gamma''_{\tau}$ can be non-zero are those that differ in at most $O(\log k)$ positions from a reduced template.

173
We further decompose $\Gamma''_\tau$ into a sum over random variables $\Gamma''_{\tau P}$, for feasible paths $P$ associated with $\tau$, where $\Gamma''_{\tau P} = 1$ if $P$ is represented in $T$, and 0 otherwise. Now, there are at most $k^j$ reduced templates with $j$ $*$'s, and hence at most $k^{O(\log k)}$ $k^j$ arbitrary templates with $j$ $*$'s for which $\Gamma''_\tau$ can be non-zero. For each such $\tau$, there are at most $d^j$ feasible paths $P$ associated with $\tau$. Each such $P$ has a probability of at most $2^{-\left(\frac{j-1}{2}\right)}$ of being represented in $T$. Summing over all $j$ gives

$$E[\Gamma''] \leq \sum_{\tau, P} E[\Gamma''_{\tau P}] \leq \sum_{j=1}^{k} k^j d^j k^{O(\log k)} 2^{-\left(\frac{j-1}{2}\right)}$$

$$\leq k^{O(\log k)} \sum_{j=1}^{k} kd \left( \frac{kd}{2(j-2)} \right)^{j-1}$$

Once $j$ is $\Theta(\log kd) = \Theta(\log \log n)$, each term is $O(1)$, so

$$E[\Gamma''] \leq k^{O(\log k)} O(\log \log n)(kd)^{O(\log \log n)}$$

$$= O(2^{O((\log \log n)^2)})$$.

We conclude with some comments on the tests used in the recovery algorithm. Recall that as we build $T$, we eliminate paths based on an internal structure check (do the edges among path nodes match those in $H$?) and a degree check (do the nodes on the path have the same degree sequence as $H$?). Although the proofs of Theorems 7.2.1 and 7.2.2 use just the internal structure check to prove uniqueness and to bound the size of $T$ respectively, it is very important in practice that the algorithm use both checks: as the experiments in the next subsection will show, one can get unique subgraphs at smaller values of $k$, and with much smaller search trees $T$, by including the degree tests. But it is interesting to note that since these theorems can be proved using only internal structure tests, the attack is robust at a theoretical level provided only that the attacker has control over the internal.
Figure 7.1: For two different choices of $d_0$ and $d_1$, the value $k = 7$ gives the attack on the LiveJournal graph a high probability of success. Both of these choices for $d_0$ and $d_1$ fall well within the degrees typically found in $G$.

structure of $X$, even in scenarios where nodes elsewhere in the graph may link to nodes in $X$ without the knowledge of the attacker. (In this case, we still require that the targeted nodes $w_j \in W$ are uniquely identifiable via the sets $N_j$, and that all degrees in $X$ remain logarithmic.)

### 7.2.3 Computational Experiments

**Social Network Data** We now describe computational experiments with the algorithm on real social network data drawn from an on-line setting. We find that the algorithm scales easily to several million nodes, and produces efficiently findable unique subgraphs for values of $k$ significantly smaller than the upper bounds in the previous subsections.
As data, we use the network of friendship links on the blogging site LiveJournal, constructed from a crawl of this site performed in February 2006. Each node in LiveJournal corresponds to a user who has made his or her blog public through the site; each user can also declare friendship links to other users. These links provide the edges of the social network we construct; they are directed, but we follow the principle of the previous subsections and convert them to undirected edges for purposes of the experiments. LiveJournal thus works well as a testbed; it has 4.4 million nodes and 77 million edges in the giant component of its undirected social network, and it exhibits many of the global structural features of other large on-line social networks. Finally, we emphasize that while LiveJournal has the right structure for performing our tests, it is not in reality an anonymous network — all of the nodes in the network represent users who have chosen to publish their information on the Web.

We simulate anonymization by removing all the user names from the nodes; we then run our attack and investigate the ranges of parameters in which it successfully identifies targeted nodes. As a first question, we examine how often $H$ can be found uniquely for specific choices of $d_0$, $d_1$, and $k$. In our construction, we generate a random external degree $\Delta_i$ for each node $x_i$ uniformly from $[d_0, d_1]$. We then create links to targeted nodes sequentially. Specifically, in iteration $i$ we choose a new user $w_i$ in $G - H$ to target; we then pick a minimal subset $X' \subseteq X$ that has not been used for any $w_j$ for $j < i$, and where the degrees of nodes in $X'$ are less than their randomly selected target degrees. We add an edge between $w_i$ and each user in $X'$. We repeat this process until no such $X'$ can be found. If, at the end of the process, some nodes in $X$ have not yet reached their target degrees, we add edges to random nodes in $G$ (and remove nodes from $W$ so that no two nodes are connected to the same subset of $X$).
Uniqueness  We say the construction succeeds if $H$ can be recovered uniquely. Figure 7.1 shows the success frequency for two different choices of $d_0$ and $d_1$ (the intervals $[10, 20]$ and $[20, 60]$), and varying values of $k$. We see that the success frequency is not significantly different for our two choices. In both cases the number of nodes we need to add to achieve a high success rate is very small – only 7. With 7 nodes, we can attack an average of 34 and 70 nodes for the smaller and larger degree choices, respectively.

We also note that the degree tests are essential for producing unique identifiability of $H$ at such a small value of $k$. In fact, each of the 734 possible Hamiltonian graphs on 7 nodes actually occurs in the LiveJournal social network, so it is only because of its degree sequence in $G$ that our constructed subgraph $H$ is unique. (Theorem 7.2.1 does guarantee that a large enough $H$ will be unique purely based on its internal structure; this is compatible with our findings since the analyzed bound of $(2 + \delta) \log n$ is larger than the value $k = 7$ with which we are succeeding in the experiments.)

Efficient Recovery  In addition to being able to find $H$ reliably, we must be able to find $H$ quickly. We argued above that the size of $T$ would remain sufficiently small that our search algorithm would be near-linear. In our experiments on the LiveJournal friendship graph we find that, in practice, the size of $T$ is not much larger than the number of nodes $u$ such that $d(u) = d(x_1)$. For instance, when $d_0 = 10$ and $d_1 = 20$, there are an average of 70,000 nodes which have $d(u) = d(x_1)$, while the size of $T$ is typically about 90,000.

Detectability  Finally, we consider the detectability of the attack. Specifically, from the point of view of the attacker, it is important that the curator of the data,
who is releasing the anonymized version, not be able to discover and remove $H$. As the curator does not have access to the secret degree sequence or the edges within $H$, they cannot employ the same algorithm the attacker uses to discover $H$. However, if $H$ were to stand out significantly in some other way, there might be an alternate means for finding it.

This is a difficult issue to capture formally, but we provide the following indications that the subgraph $H$ may be hard to discover. First is the simple fact that $H$ has only 7 nodes, so it is difficult for any of its graph-theoretic properties to stand out with much statistical significance. Second, we describe some particular ways in which $H$ does not stand out. To begin with, the internal structure of $H$ is consistent with what is present in the network. For example, we have already mentioned that every 7-node Hamiltonian graph already occurs in LiveJournal, so this means that there are already subgraphs that exactly match the internal structure of $H$ (even if not its pattern of attachment to $G$, which is also used to identify it). More generally, almost all nodes in LiveJournal are part of a very dense 7-node subgraph: If we look at all the nodes with degree at least 7, and consider the subgraph formed by those nodes and their 6 highest-degree neighbors, over 90% of such subgraphs have at least $11 > \frac{1}{2}(\binom{7}{2})$ edges. These subgraphs are also almost all comparably well-connected to the rest of $G$.

### 7.3 The Cut-Based Attack

In the walk-based attack just presented, one needs to construct a logarithmic number of nodes in order to begin compromising privacy. On the other hand, we can show that at least $\Omega(\sqrt{\log n})$ nodes are needed in any active attack that requires
a subgraph $H$ to be uniquely identifiable with high probability, independent of both the structure of $G - H$ and the choice of which users to target.

It is therefore natural to try closing this gap between the $O(\log n)$ number of nodes used by the first attack, and the $\Omega(\sqrt{\log n})$ lower bound required in any attack. With this in mind, we now describe our second active attack, the cut-based attack; it matches the lower bound by compromising privacy using a subgraph $H$ constructed on only $O(\sqrt{\log n})$ nodes. While the bound for the cut-based attack is appealing from a theoretical perspective, there are several important respects in which the walk-based attack that we saw earlier is likely to be more effective in practice. First, the walk-based attack comes with a much more efficient recovery algorithm; and second, the walk-based attack appears to be harder for the curator of the data to detect (as the cut-based attack produces a densely connected component attached weakly to the rest of the graph, which is uncommon in many settings).

**The Construction of $H$**

We begin the description of the cut-based attack with the construction of the subgraph $H$.

1. Let $b$, the number of users we wish to target, be $\Theta(\sqrt{\log n})$, and let $w_1, w_2, \ldots, w_b$ be these users. First, for $k = 3b + 3$, we construct a set $X$ of $k$ new user accounts, creating an (undirected) edge between each pair with probability $1/2$. This defines a subgraph $H$ that will be in $G$.

2. Let $\delta(H)$ denote the minimum degree in $H$, and let $\gamma(H)$ denote the value of the minimum cut in $H$ (i.e. the minimum number of edges whose deletion disconnects $H$). It is known that for a random graph $H$ such as we have constructed, the following properties hold with probability going to 1 exponentially quickly in
k [17]: first, that \( \gamma(H) = \delta(H) \); second, that \( \delta(H) \geq (1/2 - \varepsilon)k \) for any constant \( \varepsilon > 0 \); and third, that \( H \) has no non-trivial automorphisms. In what follows, we will assume that all these properties hold: \( \gamma(H) = \delta(H) \geq k/3 > b \), and \( H \) has no non-trivial automorphisms.

(3) We choose \( b \) nodes \( x_1, \ldots, x_b \) in \( H \) arbitrarily. We create a link from \( x_i \) to \( w_i \), so that the edge \((x_i, w_i)\) will appear in the anonymized graph \( G \).

**Efficiently recovering \( H \) given \( G \)**  
Now, when \( G \) is released, we identify the subgraph \( H \) and the targeted users \( w_1, \ldots, w_b \) using the following recovery algorithm.

(1) We first compute the Gomory-Hu tree of \( G \) [54, 61] — this is an edge-weighted tree \( T \) on the node set \( V \) of \( G \), such that for any \( v, w \in V \), the value of the minimum \( v \)-\( w \) cut in \( G \) is equal to the minimum edge weight on the \( v \)-\( w \) path in \( T \). Computing \( T \) is the most expensive step of the recovery algorithm, computationally. While the Gomory-Hu tree is constructable in polynomial time, it is significantly less efficient than the method employed by the walk-based attack. On the other hand, recent experiments in Web graph analysis indicate that Gomory-Hu tree computations can in fact be made to scale to very large graphs [54].

(2) We delete all edges of weight at most \( b \) from \( T \), producing a forest \( T' \). To find the set of nodes \( X \) we constructed, we iterate through all components of \( T' \) of size exactly \( k \) — let them consist of node sets \( S_1, S_2, \ldots, S_r \) — and for each such \( S_i \) we test whether \( G[S_i] \) is isomorphic to \( H \). These isomorphism tests can be done efficiently, even by brute force, since \( k! = o(n) \). Below, we prove that with high probability, there will be a single \( i \) such that \( G[S_i] \) is isomorphic to \( H \), and that
is equal to our set $X$ of new user accounts.

(3) Since $H$ has no non-trivial automorphisms, from knowledge of $S_i$ we can identify the nodes $x_1, \ldots, x_b$ that we linked to the targeted users $w_1, \ldots, w_b$ respectively. Hence we can identify the targeted users as well, which was the goal.

If we wish to target a much larger number $b$ of users, we choose a number $b' = \Theta(\sqrt{\log n})$, and we partition the targeted users into sets $U_1, U_2, \ldots, U_s$, where $s = \lceil b/b' \rceil$, and each $U_i$ except possibly the last has size $b'$. We then apply the above construction to each $U_i$, using a subgraph $H_i$ chosen independently for the attack on $U_i$ (and note that we still compromise edges between all pairs in $W = \bigcup_{i=1}^s U_i$).

Analysis of the cut-based attack  We focus on the version where $b = \Theta(\sqrt{\log n})$. The crux of the analysis is the proof of the following claim stated earlier.

**Theorem 7.3.1.** Let $T$ be the Gomory-Hu tree of $G$, let $T'$ be the forest obtained by deleting all edges of weight at most $b$, and let $S_1, S_2, \ldots, S_r$ be the node sets of all components of $T'$ that have size exactly $k$. Then with high probability, there is a single $i$ such that $G[S_i]$ is isomorphic to $H$, and the set $S_i$ is equal to $X$.

**Proof.** We first argue that $X$ appears in the list of sets $S_1, \ldots, S_r$, and to do this, it is enough to show that $X$ forms a single component in $T'$. Indeed, if $v, w \in X$ belonged to different components of $T'$, then the $v$-$w$ path in $T$ would have to contain an edge of weight at most $b$, contradicting the fact that $\gamma(H) > b$. Further, if $v \in X$ and $w \notin X$ belonged to the same component of $T'$, then the minimum $v$-$w$ cut in $G$ would have weight greater than $b$, contradicting the fact that there is a $b$-edge cut separating $H$ from $G - H$.

Thus $S_i = X$ for some $i$. We now argue that with high probability, the subgraph
\(G[S_j]\) is not isomorphic to \(H = G[X]\) for any \(j \neq i\). Let \(S_j = \{s_{j,1}, \ldots, s_{j,k}\}\), and let \(X = \{x_1, \ldots, x_k\}\). For a bijection \(f\) from \(\{1, 2, \ldots, k\}\) to itself, let \(\mathcal{E}_{j,f}\) be the event that the subgraphs \(G[S_j]\) and \(H\) are isomorphic under the mapping that sends \(s_{j,i}\) to \(x_{f(i)}\). Since the sets \(S_j\) and \(X\) are disjoint, \(\Pr[\mathcal{E}_{j,f}] = 2^{-\binom{k}{2}}\). As long as \(k \geq 1 + \sqrt{(2 + \varepsilon) \log n}\) for any constant \(\varepsilon > 0\), we have
\[
\Pr[\mathcal{E}_{j,f}] = 2^{-\binom{k}{2}} \leq 2^{-(1+\varepsilon/2)\log n} = n^{1-\varepsilon/2}.
\]

We are interested in the probability of the event \(\mathcal{E} = \bigcup_{j,f} \mathcal{E}_{j,f}\). Since there are at most \(n/k\) possible sets \(S_j\), we have
\[
\Pr[\mathcal{E}] \leq \sum_{j,f} \Pr[\mathcal{E}_{j,f}] \leq (n/k)k! \cdot 2^{-\binom{k}{2}} \leq (k-1)! \cdot n^{-\varepsilon/2},
\]
which goes to 0 with \(n\) since \(k!\) grows more slowly than \(n^\alpha\) for any constant \(\alpha > 0\) when \(k\) is \(O(\sqrt{\log n})\).

**Some specific numbers for the cut-based attack** It is useful to supplement the asymptotic results for the cut-based attack with some specific numbers. If the network \(G\) has 100 million nodes, then by creating 12 new user accounts we can succeed in identifying 3 chosen users in the system with probability at least .99. Creating 15 new user accounts leads to a microscopically small failure probability.

The calculation is as follows. We first generate 100 random 12-node graphs \(H_1, \ldots, H_{100}\), and see if any of them lacks non-trivial automorphisms and has a minimum cut of size at least 4. If any of them does, we choose one as our 12-node subgraph \(H\). Computational experiments show that a random 12-node graph will have no non-trivial automorphism, and \(\gamma(H) \geq 4\) with probability roughly 0.25. Thus, with probability well over 0.999, one of the 100 graphs \(H_i\) will have this pair of properties. Now, if we use the \(i^{th}\) of these random graphs in the construction, for
a fixed \( i \), then applying the argument and notation from the proof of Theorem 7.3.1, there are at most 8333333 possible components \( S_j \) of size 12 in the forest \( T' \), and so \( \Pr[\mathcal{E}] \leq 8333333 \cdot 12! \cdot 2^{-66} < 6 \cdot 10^{-5} \). Hence the probability that any \( H_i \) will lead to non-uniqueness when attached to \( G \) is at most .006, and so in particular this holds for the \( H_i \) that we choose as \( H \).

### 7.4 Passive Attacks

In a passive attack, regular users are able to discover their locations in \( G \) using their knowledge of the local structure of the network around them. While there are a number of different types of passive attacks that could be implemented, here we imagine that a small coalition of passive attackers collude to discover their location. By doing so, they compromise the privacy of some of their neighbors: those connected to a unique subset of the coalition, and hence unambiguously recognizable once the coalition is found.

Here, we imagine that a coalition \( X \) of size \( k \) is initiated by one user who recruits \( k - 1 \) of his or her neighbors to join the coalition. (Other structures could lead to analogous attacks.) We assume that the users in the coalition know the edges amongst themselves – the internal structure of \( H = G[X] \), using the terminology from the active attack. We also assume that they know the names of their neighbors outside \( X \). This latter assumption is reasonable in many cases: for example, if \( G \) is an undirected graph built from messages sent and received, then each user in \( X \) knows its incident edges. Other scenarios imply different levels of information: for example, if an undirected released network \( G \) is obtained from a directed graph where \((u,v)\) indicates that \( v \) is in \( u \)'s address book, then a node \( u \)
Figure 7.2: Probability of success for different coalition sizes, in the LiveJournal graph. When only the degrees and internal structure of the coalition are taken into account, a coalition of size 5 is needed to give a high probability of success. When the more refined version of the algorithm is used, and the edges connecting $H$ to $G - H$ are considered, only 4 users need collude.

does not necessarily know all its inbound edges, and hence doesn’t know its full neighbor set in the undirected graph $G$. However, in the comparably plausible variant in which the directed version of an address book network is released, the nodes in $X$ will have all the information they need for the passive attack.

This brings us to the details of the attack, which is analogous to the walk-based attack, except that the structure of $H$ occurs organically as a natural function of individuals using the system. A user $x_1$ selects $k - 1$ neighbors to form a coalition $X = \{x_1, x_2, \ldots, x_k\}$. The coalition knows whether the edge $(x_i, x_j)$ is in $G$ or not. The coalition also knows the neighbors outside $X$ of each $x_i$. Once $G$ is released, the coalition runs the same search algorithm described in the walk-based attack, with a minor modification due to the fact that $H$ need not have a Hamiltonian
path, but instead has a single node connected to all others.

To help the passive attack succeed, we can incorporate a further optimization that was not explicitly discussed earlier in the walk-based active attack experiments. For each non-empty set \( S \subseteq \{1, 2, \ldots, k\} \), we let \( g(S) \) denote the number of users to whom exactly the coalition members \( \{x_i : i \in S\} \) are connected. Using this information, a path in \( T \), corresponding to nodes \( f(\alpha_1), \ldots, f(\alpha_k = \alpha) \), must satisfy an additional constraint. If we define \( g_\alpha(S) \) analogously to \( g(S) \), but for the sequence \( f(\alpha_1), \ldots, f(\alpha_k = \alpha) \) instead of \( x_1, \ldots, x_k \), then for \( \alpha \) to correspond to a match of \( H \), it must have \( g(S) = g_\alpha(S) \), for all non-empty \( S \subseteq \{1, \ldots, k\} \).

Once the coalition \( X \) finds itself, it is able to determine the identity of some subset of its neighbors in \( G - X \). If a user \( w \) is connected to \( \{x_i : i \in S\} \), and \( g(S) = 1 \), then the identity of the user \( w \) can be uniquely recovered in \( G \). As the coalition has not specifically targeted any nodes, it is possible (and indeed likely for small coalitions) that although they can uniquely find themselves, they cannot locate any specific users other than themselves. However, empirically, we find that once a coalition is moderately-sized, it can compromise the privacy of at least some users.

Since the structure of \( H \) is not randomly generated, there is no \textit{a priori} reason to believe that it will be uniquely findable, or that the above algorithm will run efficiently. Indeed, for pathological cases of \( G \) and \( H \) the problem is NP-Hard. However, we find on real social network data that the instances are not pathological, and that subgraphs on small coalitions tend to be unique and efficiently findable.

The primary disadvantage of this attack in practice, as compared to the active
Figure 7.3: As the size of the coalition increases, the number of users in the LiveJournal graph compromised under the passive attack when the coalition successfully finds itself increases superlinearly. The number of users the semi-passive attack compromises increases exponentially.

attack, is that it does not allow one to compromise the privacy of arbitrary users. However, a natural extension is a semi-passive attack whereby a coalition of existing users colludes to attack specific users. To do this, the coalition $X$ forms as described above with $x_1$ recruiting $k - 1$ neighbors. Next, the coalition compares neighbor sets to find some set $S \subseteq X$ such that $g(S) = 0$. Then, to attack a specific user $w$, each user in $\{x_i : i \in S\}$ adds an edge to $w$. Then, assuming that the coalition can uniquely find $H$, they will certainly find $w$ as well.
7.4.1 Computational Experiments

Here we consider the passive attack on the undirected version of the LiveJournal graph. For various values of $k$, we consider a coalition of a user $x_1$, and his or her $k - 1$ highest-degree neighbors. (We also consider the case where $x_1$ selects $k - 1$ neighbors at random, and see that the success rate is similar.) We do this for a randomly chosen sample of users $x_1$ whose degree is at least $k - 1$. We then imagine that these users carry out the attack described, searching all of $G$ for a match. In our experiments, we consider both the simple version where the coalition uses only the internal structure of $H$ and the degree sequence, and also the refined version where additional structure of the connections between $H$ and $G - H$ is taken into account via the function $g(S)$.

We find that even coalitions as small as 3 or 4 users can often find themselves uniquely, particularly when using the refined version of the algorithm. Figure 7.2 summarizes the success rates for different-sized coalitions using both recovery algorithms. Furthermore, with minimal preprocessing, $G$ can be searched for a particular coalition almost immediately: On a standard desktop, it takes less than a tenth of a second, on average, to find a coalition of size 6.

At first glance, these results seem at odds with the results for the active attack in Figure 7.1, as the passive attack is producing a higher chance of success with fewer nodes. However, in the active attack, we limited the degrees of the users created in an effort to remain inconspicuous. In the passive attack, there is no such limit, and many users’ highest-degree neighbor has degree well over the limit of 60 that we imposed on the active attack. Since there are fewer users with higher degrees, this has the effect of increasing the findability of $H$. When we consider only those coalitions whose members all have degrees analogous to those in the
active attack, the results are similar to the active attack.

While the above results show that a coalition can find itself easily, this does not mean that it can identify other nodes with certainty. Clearly, a coalition of size $k$ cannot compromise more than $2^k - 1$ users, and in practice we see that the actual number is typically much smaller than this. Figure 7.3 shows the average number of users compromised by successful coalitions of various sizes. We see that even with a coalition of size 6, the number of compromised users tends to be small. However, with a semi-passive attack, we can greatly increase the number of users compromised.

Figure 7.3 shows the increased number of users typically compromised by the semi-passive attack (and recall that these users can be chosen arbitrarily by the coalition). Moreover, when the coalition is compromising as many users as possible, the semi-passive attack tends to have a higher success rate.

7.5 Discussion

It is natural to ask what conclusions about social network data should be drawn from this work. As noted at the outset, our work is not directly relevant to all settings in which social network data is used. For example, much of the research into on-line social networks is conducted on data collected from Web crawls, where users have chosen to make their network links public. There are also natural scenarios in which individuals work with social network data under safeguards that are primarily legal or contractual, rather than computational, in nature — although even in such cases, there are compelling reasons why researchers covered by contractual relationships with a curator of sensitive data should still only publicly
release the results of analyses that are carried out through a privacy mechanism, to prevent the information in these analyses from implicitly compromising privacy. In cases such as these, where computational safeguards are not the primary focus, important questions of data utility versus privacy still arise, but the questions in these cases are not something that our results directly address.

What our results do show is that one cannot rely on anonymization to ensure individual privacy in social network data, in the presence of parties who may be trying to compromise this privacy. And while one natural reaction to these results is to try inventing methods of thwarting the particular attacks we describe, we think this misses the broader point of our work: true safeguarding of privacy requires mathematical rigor, beginning with a clear description of what it means to compromise privacy, what are the computational and behavioral capabilities of the adversary, and to what information does it have access.

7.6 Related work

In a variety of settings different from the social network context here, recent work has considered ways of attacking anonymization and related schemes using content analysis of the text generated by users [13, 103], time series analysis of the time-stamps of user actions [98], or linkages among user records in different datasets [118]. In our case, however, both the passive and active attackers do not have access to highly resolved data like time-stamps or other numerical attributes; they can only use the binary information about who links to whom, without other node attributes, and this makes their task more challenging. Indeed, constructing the subgraph $H$ can be seen as a kind of structural steganography, hiding secret
messages for later recovery using just the social structure of $G$.

In this way, our approach can be seen as a step toward understanding how techniques of privacy-preserving data mining (see e.g. [15, 41, 47, 51, 95] and the references therein) can inform how we think about the protection of even the most skeletal social network data. We take up this discussion further in the final section.
CHAPTER 8
CONCLUSIONS

In the preceding chapters, we have seen a number of uses for various large-scale Web datasets. In each chapter, we explored the data from these Web sites in different ways, with differing goals. By careful design of these experiments, we were able to gain a number of insights into the systems which generated the data. In some cases, these insights were suggestive of more general principles at work, with possible implications about the principles of human behavior. In all cases, the data we worked with was on a massive scale, which allowed us to be more confident in our results, but at the same time made it important to design efficient algorithms for our experiments.

In Chapter 3 we looked at the relationship between social networks and groups. In the data we looked at, users identified themselves with various groups, and this membership grew and changed over time. By employing techniques from machine learning, we were able to extract some important features from the data which led us to observations about the impact of various network features on group evolution. We found that even in very different contexts, some of the same principles apply to group membership. While we were limited to two datasets, the similarities between them suggest that these principles are more the results of more general mechanisms. In addition, when we enriched our network data with topics, we were able to study the flow of information between groups.

In addition to studying the evolution of groups, we must also consider the evolution of the graphs in which these groups exist. In Chapter 4, we were able to examine four different networks, each of which had timestamps on all nodes and edges. This additional information enabled us to consider the evolution of the
networks in a way which would not be possible without the timestamp information. Because we knew exactly what the state of the network was at the instant each new edge appeared, we were able to quantitatively evaluate various network models. This led us to concretely show an important feature which realistic social network generation models should have – a sense of locality.

Of course, there is much more going on on the Web than social networks, and in Chapter 5 we looked at search query logs, and in particular the spatial properties of various queries. We showed how one can identify the regional focus of a query, extracting the central location of a query. Our method was successful at doing this, even on quite sparse queries, in spite of the inherent randomness in the data, and the imprecision of geolocation from IP addresses. Not only could we extract the ‘center’ for a query, but we also showed how to find the ‘spread’ of a query, discovering whether it was tightly centered, or only vaguely concentrated. In doing this, it was important to consider the scale of the data we were working with – hundreds of millions of queries a day – and the algorithms we designed were efficient enough that we were able to run them on thousands of different queries.

In Chapter 6, we looked at another dataset, with a different sort of goal. Here, rather than simply trying to understand the system generating the data, we designed an algorithm specifically to change the way the system was run. By observing the way that individuals used the system, we gained important insights into the processes governing user clicks on yahoo.com. Using these insights, we designed an algorithm to optimize content scheduling on sites like yahoo.com, which have a featured item on their home pages. While the general problem we attempted to solve was hard in the computational sense, we showed that real data had special properties which lead to computationally efficient algorithms. As a result, we pre-
sented a practical algorithm which optimally schedules content. According to our simulations, this could potentially lead to significant improvements in overall user engagement.

Finally, while we have shown some of the benefits of studying these sorts of datasets, Chapter 7 illustrated that care must be taken in these sorts of studies. If the data under examination is sensitive, it is very difficult to come up with a safe way to release it to the general research community. Time and again, companies which have attempted to do so, even if when they made some attempts to scrub the data clean, have ended up with egg on their faces. While it now seems obvious that rich, annotated data should not be released because that richness allows deanonymization, we showed that even a bare dataset of an unlabeled, undirected graph may lead to potential privacy violations. The message here is that great care must be taken whenever any data is released, because there is no way to undo such a release, and even seemingly innocuous data might be potentially compromising.

8.1 Future Work

8.1.1 Experimentation

The work in this thesis has focused mostly on explaining the data and understanding the users generating it. While this, and similar work has brought us a long way in understanding social networks, search queries, and server logs, there is only so much that can be done without actually modifying or interacting with the systems to see how they react. For instance, in our study of groups we suggested that
groups be thought of as something new diffusing in the network. Can we then use this information to try to maximize the spread of something new? For instance, if we wanted to target some group of individuals with ads, and then hope for the new idea to spread virally, we could use some of these observations, along with the algorithms in [70] to do so. Would the results match our expectations?

The same sorts of questions apply to the work in Chapter 4. We have some ideas about how networks evolve over time; can we use these to encourage robust growth? LinkedIn and Facebook have incorporated a ‘friend suggester’ feature which is one application of this sort of work. Many of the observations made in Chapter 4 seem like they could be helpful in an application like this, but they are as of yet unproven.

In Chapter 6 we were able to at least simulate our algorithms on some real data, but until we actually perform some real experiments, we cannot be sure that our assumptions are valid. There are also a number of open questions here relating to what happens if all the conditions we observe in the Yahoo! data are not met.

Unfortunately performing these sorts of applied experiments will likely remain difficult, as the number of companies with the power to do so is small, and they are understandably wary of experimental changes to live systems. Despite these difficulties, there is only so much we can do by simply observing logs files, and the next general step in the line of research embodied here is to actively influence and change these systems, hopefully for the better.
8.1.2 Multi-Modal Studies

Another important direction for these sorts of studies is to look at how different systems interact with each other. For instance, users of Flickr are also likely to use other Yahoo! products and how they use these other products is probably related to how they use Flickr. By understanding these relationships, we can seek to improve the user experience. Additionally different types of users are using the systems in different ways, and we can apply what we learn about a user in one system to how we tailor a different system to that user. We might, for example, discover something about a user’s interests from that user’s query pattern, and we could potentially use this information to customize the delivery of news.

Again though, we run into issues of user privacy, as a user might not want his searches impacting what news is displayed. It is certainly not hard to imagine the sorts of trouble this might cause. In general, this brings up the important policy considerations relating to privacy. While we would like to make Web systems as useful as possible, we must continually weigh both system design and research considerations against the expectations of our users that we respect their privacy.

8.1.3 Scalability

Another important broad goal is to develop algorithms which can scale to the types of datasets being generated today. With datasets often well into the terabytes, it is clearly important to have fast algorithms. In many cases, accuracy is a smaller concern than speed, and solutions which are nearly correct are acceptable. Additionally, cheap hardware has made distributed computing the norm at many companies, and algorithms which can be easily distributed are much more
practical, even if they take much more overall computation.

While there are many problems in this realm, the increasing prominence of social networks has made algorithms capable of operating on billions of edges particularly important. One direction which remains relatively unexplored but holds some promise is to use the properties of the social networks in the design of algorithms for classical problems. For instance, we learned in Chapter 4 that most edges are local. Can we use that information to store networks in an efficient way on disk to minimize the number of disk reads for something like shortest path computation? Graph problems like this, which are in some sense solved for smaller graphs, are still challenging at this scale.


[43] P. Domingos, M. Richardson. Mining the Network Value of Customers. Proc. 7th Intl. Conf. Knowledge Discovery and Data Mining,


[70] D. Kempe, J. Kleinberg, E. Tardos. Maximizing the Spread of Influence
through a Social Network. Proc. 9th ACM SIGKDD Intl. Conf. on Knowledge
Discovery and Data Mining, 2003.

SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, 2002.

[72] J. Kleinberg. Temporal dynamics of online information streams. In M. Garo-


32nd ACM Symposium on Theory of Computing, 2000

[75] G. Kossinets and D. J. Watts. Empirical Analysis of an Evolving Social Net-
work. Science 311:88–90, 2006

[76] D. Krackhardt and M. Handcock. Heider vs. Simmel: Emergent features in
dynamic structure. In Statistical Network Analysis: Models, Issues, and New


[80] R. Kumar, R. Novak, P. Raghavan, A. Tomkins. Structure and evolution of
blogspace. CACM, 47(2004).

[81] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E.


