MODELS AND ALGORITHMS FOR EVENT-DRIVEN NETWORKS

BY BRIAN EVAN THOMPSON

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

S. Muthu Muthukrishnan

and approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

January, 2014

ABSTRACT OF THE DISSERTATION

Models and Algorithms for Event-Driven Networks

by Brian Evan Thompson

Dissertation Director: S. Muthu Muthukrishnan

Many real-world systems can be represented as networks driven by discrete *events*, each event identified by the time at which it occurs and the parties involved. An event could be a meeting, a stock trade, a phone call, an email, a gang fight, an online or off-line purchase, a blog post, a conference, or the transmission of an IP packet. Innovations in technology have increased our ability to collect massive amounts of digital data from such networks, which presents both new opportunities and new challenges. In this work, we develop new theoretical models and efficient algorithms that leverage the temporal and relational information inherent in the data to better understand and analyze real-world networks. In particular, we consider three problems: (1) detecting correlated events in communication networks; (2) discovering functional communities; and (3) modeling collaboration in academia.

First we present a new stochastic model for event-driven networks, and with it develop two algorithms – a streaming local algorithm, and an efficient global algorithm – to detect statistically correlated activity. We demonstrate that our approach, which models each communication channel as its own stochastic process, is better able to accommodate the temporal variability present in real-world communication networks than existing methods.

Next we study diffusion processes in information networks, identifying functional communities as groups of individuals who participate in the dissemination of common content by

reframing the problem as one of co-clustering sparse matrices. We propose a new co-clustering algorithm that does not require user-specified parameters, and leverages sparsity in the data to run in sublinear time in the size of the matrix.

Finally, we build a game-theoretic model for academic collaboration, representing the academic environment as a repeated game in which each researcher tries to maximize his or her academic success. We find analytically that limitations of existing collaboration models may result in misleading predictions about people's behavior.

Funding Support

Part of this work was supported by the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. DE-AC52-07NA27344.

Part of this work was supported by the U.S. Department of Homeland Security under grant number 2008-ST-104-000016. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security.

Part of this work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-706. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

Acknowledgements

Both this dissertation and I are the product of a great many influences, and I am truly grateful for the people who have supported me along this path, and who have shaped me both emotionally and intellectually. I have been incredibly fortunate to have so many mentors and role models throughout the years, without whose inspiration and guidance I would certainly not be who I am today.

First, I would like to thank my advisor, Muthu, for taking me on as a student so late in my graduate career, and for giving me guidance while still allowing me the freedom to pursue, to struggle, and to learn from my mistakes.

To my former advisor, Danfeng Yao, thank you for guiding me during my early years and pushing me into the research world.

Thank you also to the other members of my committee: Rebecca Wright, Paul Kantor, and Hanghang Tong. I think that your feedback has resulted in a stronger dissertation, and has given me greater vision in understanding how my work fits together, and where it fits in the greater scheme of things.

My sincere thanks to Fred Roberts, Tami Carpenter, Tina Eliassi-Rad, and James Abello, for your mentorship. I have learned a great deal from each of you.

Thanks also to my other collaborators and mentors at Rutgers, DIMACS/CCICADA, ACS, LLNL, HP Labs, and elsewhere.

I would be remiss not to acknowledge the DHS Fellowship which funded me for three years, and also the wonderful Computer Science Department and DIMACS staff, for providing the resources and environment I needed to pursue my studies.

And last but not least, thank you from the bottom of my heart to my family and friends, for nurturing my curiosity, giving me your boundless encouragement and support, and tolerating me.

Dedication

This dissertation is dedicated to my dear and loving parents. I will never be able to fully appreciate everything that you have given me. Thank you.

Table of Contents

Abstract ii

Funding Support iv

Acknowledgements v

Dedication vi

1. Introduction 1

1.1. Motivation 1

1.2. Overview and Contributions 2

1.2.1. Detecting Correlated Events in Communication Networks 3

1.2.2. Discovering Functional Communities 4

1.2.3. Modeling Collaboration in Academia 5

1.3. Outline of the Dissertation 6

2. Background 7

2.1. Definitions and Framework 7

2.1.1. Graphs 7

2.1.2. Event-Driven Networks 8

2.2. Themes and Survey of Literature 9

2.2.1. Graph Analysis 9

2.2.2. Temporal Dynamics 10

2.2.3. Group Behavior 11

2.2.4. Attribution 13

2.2.5. Computational Realizability 14

2.3. Connection to Projects 16

2.3.1. Detecting Correlated Events in Communication Networks 16
2.3.2. Discovering Functional Communities 17
2.3.3. Modeling Collaboration in Academia 18

3. Detecting Correlated Events in Communication Networks 19
3.1. Introduction 19
3.1.1. Background and Motivation 19
3.1.2. Related Work 20
3.1.3. Contributions and Outline 22
3.2. Methodology 22
3.2.1. The REWARDS Model 23
3.2.2. Measuring Statistical Correlation 23
3.2.3. Recency 25
3.2.4. Correlated Event Detection 27
3.2.5. Complexity Analysis 30
3.3. Evaluation 32
3.3.1. Modeling Inter-Arrival Times 32
3.3.2. Robustness to Time Scale 33
3.3.3. Accuracy and Precision 37
3.3.4. Detection Latency 38
3.3.5. Scanning Activity in IP Traffic 39
3.3.6. Physical Proximity 40
3.4. Discussion 42
3.4.1. Extensions and Applications 42
3.4.2. Limitations of Our Approach 43
3.4.3. Significance and Impact 44

4. Discovering Functional Communities 46
4.1. Introduction 46
4.1.1. Background and Motivation 46

4.1.2. Related Work . . . 47
4.1.3. Contributions and Outline . . . 49
4.2. Methodology . . . 50
4.2.1. Preliminaries . . . 50
4.2.2. Choosing a Metric . . . 51
4.2.3. The CC-MACS Algorithm . . . 55
4.2.4. Complexity Analysis . . . 55
4.3. Evaluation . . . 61
4.3.1. Prediction Accuracy . . . 62
4.3.2. Finding Block Structure . . . 63
4.3.3. Discovering Functional Communities in Social Media . . . 64
4.4. Discussion . . . 65
4.4.1. Extensions and Applications . . . 65
4.4.2. Limitations of Our Approach . . . 66
4.4.3. Significance and Impact . . . 66

5. Modeling Collaboration in Academia . . . 68
5.1. Introduction . . . 68
5.1.1. Background and Motivation . . . 68
5.1.2. Related Work . . . 69
5.1.3. Contributions and Outline . . . 70
5.2. Methodology . . . 71
5.2.1. Preliminaries . . . 71
5.2.2. Game-Theoretic Model . . . 73
5.3. Evaluation . . . 76
5.3.1. Collaboration Model . . . 76
5.3.2. Single-Player Game . . . 79
5.3.3. Two-Player Game . . . 80
5.3.4. Multi-Player Game . . . 83

5.4. Discussion 85
5.4.1. Extensions and Applications 85
5.4.2. Limitations of Our Approach 86
5.4.3. Significance and Impact 87

6. Related Problems and Future Work 88
6.1. Measuring Pairwise Influence 88
6.2. Innovation and Circulation 89
6.3. Cascade Partitioning 91

7. Conclusions 93
7.1. Discussion of Themes 93
7.1.1. Graph Analysis 93
7.1.2. Temporal Dynamics 94
7.1.3. Group Behavior 94
7.1.4. Attribution 95
7.1.5. Computational Realizability 95
7.2. Summary of Contributions 96
7.3. The Big Picture 98
References 100

Chapter 1

Introduction

1.1 Motivation

Many real-world systems can be represented as networks driven by discrete *events*, each event identified by the time at which it occurs and the parties involved. An event could be a meeting, a stock trade, a phone call, a gang fight, an online or off-line purchase, a blog post, a conference, or the transmission of an IP packet. Innovations in technology have increased our ability to collect massive amounts of digital data from such networks, which presents both new opportunities and new challenges. The goal of our research is to develop techniques that leverage the temporal and relational information inherent in the data to better understand and analyze these networks.

Event-driven networks are relevant to a variety of real-world domains. In *communication networks*, individual people or computers communicate with one another directly through channels such as email, telephone, SMS, instant messaging services, face-to-face encounters, or IP connections. An *information network* consists of people or organizations that can both receive and broadcast information, facilitating the transfer of content across the network; examples include web logs (blogs), microblogging services such as Twitter, and other forms of social media. In a *co-occurrence network*, an event corresponds to terms occurring in the same published document. In a *co-participation network*, events correspond to individuals participating in the same activity, such as running a marathon or attending a theater performance or sporting event. Stores or markets can be seen as *purchase networks*, where an event indicates that a consumer has purchased a particular product. A *recommendation network* allows individuals to endorse an organization, item, or service, e.g. by reading an article or by "liking" a page on Facebook. The diversity of possible applications leads to networks with vastly differing properties and network structures.

Networks can be understood at multiple scales. A local perspective could look at an individual's behavior: amount of activity, type of activity, how the individual relates to others, or when new connections are formed. A global perspective could look at over-arching trends, information diffusion, formation of communities, or changes in network structure.

Some algorithms and data mining approaches are parameter-dependent or require domain- or network-specific knowledge. For example, when studying temporal dynamics, the granularity of analysis (e.g. aggregating or analyzing data over each second, minute, hour, day, week, or month) may significantly affect the results. A decay model requires parameters to dictate the rate of decay. Data mining approaches such as k-means clustering or k-nearest neighbors are similarly affected by the choice of parameters.

In addition, some methods of network analysis depend on a multitude of meta-data such as geospatial information or textual attributes. Such approaches would be inhibited in a context where such data is limited or not easily accessible.

In this dissertation, we aim to develop methods that:

- apply to a broad class of dynamic networks
- consider both local and global aspects of network structure and behavior
- require few or no parameters
- have minimal data requirements

In the next section, we provide an overview of three problems that arise in the analysis of real-world networks, and summarize our contributions in addressing those problems.

1.2 Overview and Contributions

In this dissertation, we develop new theoretical models and efficient algorithms to analyze temporal and behavioral aspects of real-world networks. First, we formalize a new framework for what we call *event-driven networks*. Next, we apply this framework to address three problems:

1. Detecting correlated events in communication networks – Find connected areas of the network with a high concentration of recent activity.

2. Discovering functional communities – Identify groups of individuals who participate in the dissemination of similar content.

3. Modeling collaboration in academia – Study how coauthorship relationships are shaped by researchers' individual goals.

We introduce each of these problems below.

1.2.1 Detecting Correlated Events in Communication Networks

Entities in dynamic networks often exhibit correlated behavior, which may be due to influence, environmental effects, or response to common external stimuli. However, in many scenarios these dependencies are not explicitly known. Algorithms to discover these latent correlations have applications to computer network security, intelligence, marketing, knowledge discovery, recommendation systems, and other domains. In Chapter 3 we propose a new approach to detecting correlated activity in the context of communication networks.

Since the times and rates of communication may vary across the network, many approaches for analyzing such data first aggregate communication activity over time blocks of globally-determined length. With this more uniform representation, behavior of different entities can be compared across the network using well-known time series analysis methods or other tools. However, this preliminary aggregation step may also hide correlated activity that is not visible on the time scale determined by the global parameters. In this work, we propose a new approach to correlated event detection that is able to better accommodate the temporal variability present in communication networks.

Contributions

We first present a new stochastic model for dynamic networks using tools from Renewal Theory, called the REWARDS (REneWal theory Approach for Real-time Data Streams) model. This approach aims to address the challenges of analyzing networks containing individuals with vastly different temporal and behavioral characteristics. In particular, it moves away from predominantly-used approaches that require an aggregation step or use a decay model with global parameters, which are sensitive to the time scale used for analysis. Using the REWARDS approach to model communication between each pair of nodes, we develop statistical methods to identify dependencies in the system. We validate the effectiveness and robustness of our

approach on synthetic data, and then apply it to detect correlated events in real-world email, IP traffic, and physical proximity networks.

Our main contributions can be summarized as follows:

- The REWARDS model, a streaming stochastic model for systems of point processes
- A formal definition of recency for renewal processes that is time scale-invariant
- A statistical method for measuring correlation between entities in a network that addresses the variety of temporal characteristics present in real-world networks
- A streaming local algorithm for detecting correlated activity among a fixed set of nodes
- An efficient global algorithm that simultaneously detects subsets of nodes exhibiting correlated activity in disparate parts of the network

1.2.2 Discovering Functional Communities

Community discovery is a natural task that arises in the study of social networks, but finding a mathematical formulation which captures the intuitive notion of community is an active research area. Most of the existing literature frames community discovery as the task of clustering a social network graph so that well-connected vertices are in the same cluster. When the network also serves as a medium for the dissemination of information, however, graph structure alone does not tell the whole story, since the existence of a social link does not imply information transfer.

In Chapter 4 we present an alternative approach to community discovery that identifies communities as groups of individuals who have similar behavioral patterns with respect to the dissemination of information. We do this by looking at *memes*, sets of messages with related content. Our goal is to identify groups of individuals who participate in the dissemination of multiple common memes.

Contributions

Given a set of memes from an information network, we first construct a binary matrix, where the rows correspond to individuals in the network, the columns correspond to memes, and a 1-entry indicates that the individual participated in that meme. We then frame the problem as one

of matrix co-clustering, simultaneously clustering the rows and columns of a matrix to reveal hidden structure. Driven by the goal of community discovery, we suggest that large, dense blocks, or *biclusters*, correspond to functional communities of individuals who participate in many of the same memes. We observe that existing co-clustering metrics are not designed to reward such structure, and propose a class of metrics that do. Finally, we present the CC-MACS (Co-Clustering via Maximal Anti-Chain Search) algorithm, a new heuristic algorithm which efficiently searches the space of possible co-clusterings for one which maximizes the value of a given metric.

Our main contributions can be summarized as follows:

- Two intuitive properties of co-clustering metrics that aim to reward large, dense biclusters
- A class of metrics which uniquely satisfy those properties among known metrics
- The CC-MACS algorithm, an efficient heuristic algorithm to find a good co-clustering in time sub-linear in the size of the matrix for sparse matrices

1.2.3 Modeling Collaboration in Academia

Across academic disciplines, it is natural to want to measure the impact of an individual and his or her work. Consequently, many metrics have been proposed, based on properties of an individual's research output. These are used to compare researchers to one another, influencing decisions around hiring and promotions. Such metrics start with simple counts of papers published or citations received, and become progressively more complex. However, while most metrics proposed in the literature are based on individual accomplishments, much scientific and academic progress is the result of collaborative efforts.

In this work, we aim to understand the mechanisms underlying academic collaboration. Using tools from the field of Game Theory, we study how collaboration may arise as the result of interplay between reseachers' individually-motivated behaviors.

Contributions

We begin by building a model for how researchers collaborate and how collaboration affects the number of citations a paper receives, supported by observations from a large real-world

publication and citation dataset. Using this model, we study researchers' collaborative behavior over time under the premise that each researcher wants to maximize her academic success in terms of both the quality and quantity of her research output.

Our main contributions can be summarized as follows:

- A game-theoretic framework modeling academic collaboration as a repeated game
- Formal analysis of collaboration strategies and game equilibria

1.3 Outline of the Dissertation

In Chapter 2, we lay the groundwork for this dissertation. We begin with a theoretical framework for modeling networks, including both traditional tools from Graph Theory and a new model of our own construction. We then introduce several themes that arise in the study of real-world networks and survey the relevant literature. Finally, we examine each of the three problems addressed in this dissertation, exploring the applicability of the network models and discussing relevance to the themes.

Chapters 3, 4, and 5 focus on the three problems of detecting correlated events in communication networks, discovering functional communities, and modeling collaboration in academia, respectively. For each problem, we formally define the problem, survey related work, present our methodology for solving the problem, and evaluate our approach through analysis of simulated and real-world networks.

In Chapter 6, we explore several other problems for which our framework and methodologies might be useful, and suggest directions for future work.

Chapter 7 provides additional reflection on the material presented in this dissertation, including further discussion of the central themes of our work and a summary of our contributions. We conclude with a big-picture perspective on the current state of network research and a path forward.

Chapter 2

Background

2.1 Definitions and Framework

In scientific discussion, the terms *graph* and *network* are often used interchangeably. However, while a graph is a well-defined mathematical construct, there is no single definition of network that is used consistently in the literature. We begin with some definitions from the field of Graph Theory, and then lay out a formal framework for a particular class of networks, which we call *event-driven networks*. In this dissertation, we will apply both of these constructs in our study of real-world networks.

2.1.1 Graphs

A *graph* $G = (V, E)$ is defined by a set of objects $V = V(G)$ called *vertices* and a set of object pairs $E = E(G)$ called *edges*. If the pairs are ordered, i.e. $E \subseteq V \times V$, then we say G is *directed*; if they are unordered, i.e. $E \subseteq \binom{V}{2}$, then we say G is *undirected*. A *weighted graph* is a graph G with a function w assigning weights to the edges of G, $w : E(G) \to \mathbb{R}$.

A *subgraph* H of a graph G, denoted $H \subseteq G$, is a set of vertices and edges contained in G that itself is a graph, i.e. $V(H) \subseteq V(G)$, $E(H) \subseteq E(G)$, and $(v, v') \in E(H) \Rightarrow v, v' \in V(H)$. If G is a weighted graph with weight function w, then H is a weighted graph with weight function w restricted to $E(H)$. If $V(H) = V(G)$, then H is said to *span* G.

For an undirected graph G, two vertices $v, v' \in V(G)$ are said to be *adjacent* if $(v, v') \in E(G)$; and the *neighborhood* of v, denoted $N(v)$, is the set of vertices that are adjacent to v. For a directed graph G, the *incoming neighborhood* of v is defined as $N^+(v) = \{v' \in V(G) : (v', v) \in E(G)\}$, and the *outgoing neighborhood* as $N^-(v) = \{v' \in V(G) : (v, v') \in E(G)\}$.

2.1.2 Event-Driven Networks

We define an *event-driven network* $\mathcal{N} = (\mathcal{U}, \mathcal{E})$ to consist of a set of *nodes* $\mathcal{U}$ and a set of time-stamped *events* $\mathcal{E} \subset 2^{\mathcal{U}} \times 2^{\mathcal{U}} \times \mathbb{R}^+$, each event $\varepsilon \in \mathcal{E}$ corresponding to a set of source nodes $S_\varepsilon \subseteq \mathcal{U}$, a set of recipient nodes $R_\varepsilon \subseteq \mathcal{U}$, and a time of occurrence $t_\varepsilon \in \mathbb{R}^+$. The source set for an event must be non-empty; there is no restriction on the set of recipients.

For any set of events $\mathcal{E}' \subseteq \mathcal{E}$, we can construct a graph $G_{\mathcal{E}'} = (V, E)$, with $V \subseteq \mathcal{U}$ corresponding to the network nodes that are a source or recipient of any event in $\mathcal{E}'$, and $E \subseteq V \times V$ corresponding to node pairs (u, u') such that u is a source and u' is a recipient for some event in $\mathcal{E}'$. We refer to $G_{\mathcal{E}'}$ as *the graph induced by* $\mathcal{E}'$.

For each node $u \in \mathcal{U}$, let $\mathcal{E}_u \subseteq \mathcal{E}$ denote the set of events with u as a source:

$$\mathcal{E}_u = \{\varepsilon : \varepsilon \in \mathcal{E},\ u \in S_\varepsilon\}.$$

For each node pair $(u, u') \in \mathcal{U} \times \mathcal{U}$, let $\mathcal{E}_{(u,u')} \subseteq \mathcal{E}$ denote the set of events for which u is a source and u' is a recipient:

$$\mathcal{E}_{(u,u')} = \{\varepsilon : \varepsilon \in \mathcal{E},\ u \in S_\varepsilon,\ u' \in R_\varepsilon\}.$$

For any set of events $\mathcal{E}' \subseteq \mathcal{E}$ there is a corresponding *time sequence* $T_{\mathcal{E}'} = \{t_\varepsilon : \varepsilon \in \mathcal{E}'\}$ under the natural ordering of the reals. For simplicity of notation, we let $T_u = T_{\mathcal{E}_u}$ and $T_{(u,u')} = T_{\mathcal{E}_{(u,u')}}$. Figure 2.1(a) shows the time sequences for all nodes in an event-driven network. Figure 2.1(b) shows the (non-empty) time sequences for pairs of nodes in an event-driven network.

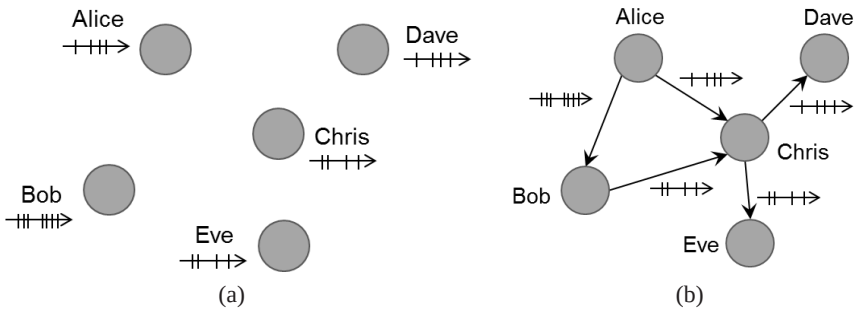


Figure 2.1: Visualization of the time sequences for all (a) nodes and (b) pairs of nodes in an event-driven network.

We note that in event-driven networks, as opposed to graphs, persistent relationships between nodes need not be defined explicitly. In Section 2.3, we discuss how these two models can be applied to real-world networks. First, we explore several themes of our work.

2.2 Themes and Survey of Literature

The primary motivation for studying real-world networks is to enable better-informed future decisions. These decisions may be based on an understanding of the current state of the network, predictions of the future state, or strategies for manipulating the network itself. We explore several themes and discuss their relevance to the study of real-world networks.

2.2.1 Graph Analysis

One approach to network analysis begins by constructing a graph G encapsulating knowledge of the network. For one example, consider a social network: $V(G)$ could be the users of the network, and $E(G)$ could be the pairs of users with an explicit social relationship in the network (e.g. friend, follower, or co-worker). This is called a *snapshot graph*, since it represents the network at a single snapshot in time. For another example, consider a phone network with a log of the sender, recipient, and time of each call. One could form a weighted, directed graph G by defining $V(G)$ to be all phone numbers that have ever been active, $E(G)$ to be all ordered pairs of numbers (a, b) such that a has called b at least once, and $w(a, b)$ to be the number of such calls. This is called a *summary graph*, since it summarizes activity in the network over a period of time. Graphs can be applied in many different contexts to represent objects and the relationships between them.

Once a graph has been constructed, a multitude of graph algorithms can be applied. Minimum-cut algorithms evaluate how robust the graph is to the severence of edges. Spanning tree algorithms can help identify a backbone structure of the network. Other algorithms could look for subgraphs with a particular structure, such as star-like formations or large cliques of pairwise-adjacent vertices.

Centrality measures aim to identify vertices of interest by quantifying the relative importance of each vertex's role in the network. Degree centrality looks for vertices with many

neighbors. Closeness centrality ranks vertices based on their distances to all the other vertices. Betweenness centrality identifies vertices which lay along shortest paths between many other vertex pairs. Eigenvector centrality is defined to recursively favor vertices with many connections to other central vertices.

Other work studies properties of real-world networks, and proposes theoretical models to describe or explain empirical observations. Well-known results state that many real-world networks have a heavy-tailed degree distribution [23, 12], small diameter [96, 7], and high clustering coefficient. Generative models have been suggested to construct graphs that exhibit some or all of these properties, such as the Watts-Strogatz model [96], the Barabási-Albert model [12], and Kronecker graph models [63].

2.2.2 Temporal Dynamics

As the above examples demonstrate, static graph analysis can provide useful information about a network. However, many real-world networks change over time. As a result, static analysis could yield misleading information for several reasons, among them: new data may have been non-existent or unavailable when the computation was performed; old information may no longer be accurate; and temporal dependencies, such as trends or periodicity in the data, may be overlooked. Additional data may make the analysis more robust or comprehensive, fill in missing or out-dated information, and decrease the chance of error.

There are many approaches to network analysis with a temporal element. One such approach is time series analysis. A *time series* describes how a scalar quantity changes over time. For example, one could track the size of the network, the degree of a node, or the number of messages sent across a link as time passes. Formally, a time series is a real-valued function f parameterized by time, and can be either continuous ($f : \mathbb{R}^+ \to \mathbb{R}$) or discrete ($f : \mathbb{Z}^+ \to \mathbb{R}$). In practice, however, it is often impossible or impractical to observe and record the functional values at all points in time. Instead, techniques are employed to more concisely represent the time series, such as sampling, fitting to a parameterized model, or low-dimensional approximation. A wide variety of representations exist in the literature, including discrete Fourier transforms [4, 31], wavelets [18, 21], piecewise functional approximations [33, 53, 19], singular value decomposition [58, 81], and symbolic approaches [68]. [26] provides a survey and

comparison of existing time series models.

Another approach models networks as *time-evolving graphs*, graphs whose vertex and edge sets change over time. In a *graph sequence* representation, the graph changes at discrete points in time, resulting in a finite sequence of static graphs which are then analyzed collectively. Alternatively, updates to the graph could occur in continuous time, usually with fewer changes occurring at any one moment. Previous work has studied real-world networks through the lens of time-evolving graphs, observing properties such as densification [64], preferential attachment [47], triadic closure [40, 80, 37], and shrinking diameter [64]. Change detection algorithms flag times at which significant changes in graph structure occur [91, 41]. Machine learning and other techniques attempt to predict the formation of new links [69]. Diffusion models are used to analyze the flow of information through a network [83, 27].

One general framework for temporal analysis is *simulation*, which entails developing a model of a real-world system, and then playing out how the system behaves over time according to the model. Simulations can be deterministic or stochastic. Network simulations may model the addition or deletion of nodes or links, interactions between nodes, and changing node or link attributes. Several general approaches have been considered in the literature, including event-based, activity-based, and process-based models [78].

Game theory can also be applied to study temporal aspects of networks. A *game* consists of rational players whose actions collectively determine the outcome of the game. Each player assigns a value to each possible outcome, and each player's goal is to arrive at an outcome of maximal value. In a *sequential game*, players alternate taking actions, which may depend on the current game state and on knowledge of other players' previous actions. In a *repeated game*, the same game is played multiple times, and players' strategies may be based on what happened in previous iterations. Applying game theoretic analysis to a network context, the players may represent nodes, and actions could be interactions between nodes.

2.2.3 Group Behavior

Network analysis would be greatly simplified if the network were modeled as a collection of objects acting independently, each of whose behavior is determined solely by information specific to that object. However, in many real-world networks an object's behavior is affected by its

relationships to and interactions with other objects. Sometimes these relationships are known and interactions observed; other times they are not, for example due to covert communication or response to common external stimuli. In either case, models which account for these higher-level correlations have the potential to provide more accurate analysis. In addition, identifying groups of nodes with similar properties or behavior can avoid redundant computation, help characterize nodes and their roles in the network, or assist with entity resolution.

One approach to identifying similarities or correlations between nodes is time series *pattern matching*. The goal is to recognize whether two nodes have exhibited similar behavior as reflected in time series corresponding to their respective activity. Techniques have been developed to detect common patterns even in the presence of shifting, distortion, missing data, or noise [87, 66].

Clustering is the task of grouping a set of objects such that elements of the same group are more similar than elements of different groups. Many different measures of similarity have been proposed, depending on the characteristics of the data and the goal of the analysis. Two general notions of similarity for clustering nodes in a network are *connectivity* and *role*. The former is based on how well-connected the nodes are; e.g. small pairwise distances, or high density of links within a group. The latter is based on nodes having similar functionality or local graph structure; they need not be in close proximity in the network graph. Given a similarity measure, many different clustering algorithms can be applied. Traditional clustering algorithms partition the objects into disjoint subsets; alternative clustering models may allow an object to belong to multiple groups or have fractional group membership, sometimes referred to as *soft clustering*.

The term *community discovery* is sometimes used to refer to connectivity-based clustering of nodes in a network, usually implying interaction or coordinated functioning among group members. *Community evolution* studies how group membership changes over time. The number of communities or the labels identifying them may be fixed, or one could allow for the formation of new communities and dissipation of existing ones. Regardless of the specific model used, there are conceptual challenges in defining and characterizing communities; for example, the term “community” has a connotation of perpetuity, yet membership is fluid and may change considerably over time.

2.2.4 Attribution

One guiding concept when applying algorithms to real-world problems is *interpretability*, how easily the results can be understood and thus applied to future decision-making. One way of making results more interpretable is through *attribution*, connecting an outcome with its source or cause.

In social psychology, attribution describes the mechanism by which people associate observed behaviors or events with causal factors. For example, somebody overhearing a verbal fight might attribute it to the participants' aggressive personalities, bad moods, or a recent misunderstanding. Literary attribution is the study of ascribing historical works of literature to a particular author. In copyright law, producers of new work must give proper attribution to ideas which are not their own. Research publications are also expected to cite others' work as a form of attribution.

Attribution is closely tied to *accountability*, the ability to hold individuals responsible for their actions. This idea is central to many legal systems, and has seen renewed interest with regard to security and privacy in online systems. The goal is to design a system that incentivizes good behavior not by directly enforcing it, but by tying bad behavior to undesirable consequences. One example is a *reputation system*, where users are assigned scores based on opinions or feedback from others, resulting in rewards or punishments, such as the granting or revocation of privileges [82].

Another related concept is *causality*. In the simplest sense, an action is said to *cause* an outcome if the outcome is a direct consequence of the action. If such a causal relationship is observed, the outcome can be attributed to the action. In many real-world scenarios, however, an outcome is the result of a multitude of factors, and causal relationships are not explicitly known; defining causality in such circumstances is the subject of debate amongst scientists [35] as well as philosophers [8, 44].

When causal relationships can not be directly observed, attribution may be determined by studying *influence*,[1] the indirect effects of one's actions. Influence can be measured by comparing the outcome when an action is performed with the outcome when the action is

[1]We use the term "influence" to include such notions as probabilistic causation and Granger causality.

not performed, all other factors remaining equal. Since we can not physically observe the outcomes under both scenarios simultaneously, assumptions are made to limit the other factors being considered, thus enabling influence to be measured empirically. For example, one may assume a closed system with a fixed set of variables, and then measure the influence of each variable by holding the others constant. This typically requires either the ability to design control experiments or a large and varied dataset that contains information about actions and the corresponding outcomes.

Since it is often difficult to formally establish causality or influence in practice, many approaches in the statistics literature focus on the weaker notion of *dependence*. Two random variables are said to be dependent if the probability distribution of values for one is different when conditioned on the value of the other. Given a set of empirical data, statistical tests can be performed to evaluate whether an action and an outcome are dependent, and therefore whether the outcome should be partially attributed to the action.

2.2.5 Computational Realizability

Many real-world networks are massive, not only with regard to the number of entities in the network, but also the volume and rate of activity. There are an estimated 2.2 billion email users worldwide, who send 45 billion emails – not including spam – every day. The World Wide Web consists of hundreds of millions of websites; the online social network Facebook boasts over a billion active monthly users; and micro-blogging site Twitter saw an average of 175 million tweets per day in 2012.[2] There are many challenges to performing network analysis on such a large scale.

Many questions that arise when analyzing group behavior in networks are *computationally hard*; that is, the time it takes to solve the problem increases very rapidly with respect to the size of the network. This is not surprising since the number of possible subsets of nodes is exponential in the size of the network. However, in many real-world settings, analysis is not helpful unless the results are found in a timely manner. Since exact solutions can not be found efficiently, approximation algorithms or heuristics may be developed to provide practical results

[2]Statistics taken from the website `Pingdom.com`, posted on their Tech Blog on January 16, 2013, compiled from various sources.

that address the real-world needs.

In addition, many network systems have physical limitations on the amount of storage space available, yet data continues accumulating over time. Even if there is space to store all of the data, it may be too cumbersome to process the entire dataset every time new data arrives. One solution is to use *streaming* algorithms, which process new data only once as it arrives, maintaining only a limited amount of information about all previous data [6].

Another approach for dealing with the great volume and rate of data is *distributed computing*. A distributed architecture stores data in multiple places rather than in one single repository. In distributed algorithms, different segments of the data are analyzed separately. If desired, the individual results may then synthesized to arrive at a final combined result.

A further challenge to real-world network analysis is *data accessibility*; that is, some data that would be helpful for analysis may not be available as desired. For example, the amount or type of data that can be collected may be limited due to physical constraints or imperfections in the data collection mechanism. A sensor may give inaccurate readings due to environmental noise. Lack of precision in data collection instruments may yield data with poor granularity. Faulty devices may result in missing data. These practical issues have prompted the development of methods that are robust to noisy or missing data. Furthermore, even after data has been recorded, there may be restrictions on its use due to privacy policy or legal concerns. The *secure multi-party computation* paradigm has been suggested as a way of addressing such concerns [99].

Finally, some computation may be subject to limitations of human observation and understanding. This is of relevance when the network is intended to model human beliefs or behaviors that are subjective, difficult to observe, or not quantifiable, or when the efficacy of a computational result is dependent on the ability of human actors to understand and respond to it.

In this dissertation, we propose new methods for analyzing real-world networks. We look at three problems that arise in the study of group behavior in networks, paying special attention to temporal aspects of the data. We develop new models and efficient algorithms to address these problems and explore the use of attribution in improving the interpretability of the results.

2.3 Connection to Projects

Here we discuss how the graph theoretic and event-driven network models can help address each of the three problems addressed in this dissertation, and explore how they relate to the themes introduced in the previous section.

2.3.1 Detecting Correlated Events in Communication Networks

In a communication network, individuals communicate with one another directly through channels such as email, telephone, SMS, instant messaging services, or IP connections. Such networks are frequently modeled as graphs, where two vertices are connected by an edge if the corresponding individuals ever interact with one another. The graph may be weighted by the frequency or volume of the interactions. We suggest that real-world communication networks can be further understood and analyzed using our event-driven network model. In Chapter 3, we explore both of these approaches.

In particular, we focus on the task of correlated event detection. We describe a set of communication events as correlated if their collective relational and temporal characteristics differ significantly from the expectation if the processes generating them were acting independently. Such correlations may indicate that the behavior of certain individuals is linked, or that the set of events in question were triggered by a common external source.

Attribution plays an important role in real-world correlated event detection. For many security applications, the utility of an anomaly detection method depends on its ability to identify sources of malicious behavior so they can be neutralized. In the context of data mining, proper attribution can lead to a better understanding of the system and more informative analysis. Our analytical methods are designed with attribution in mind; that is, in addition to simply detecting that correlated activity has occurred, our goal is to pinpoint the exact individuals and events that are responsible for that activity.

Many real-world networks of interest contain thousands or millions of nodes, with a high rate of communication, posing a computational challenge. To exascerbate the problem, information may be time-sensitive, so that results are only useful if obtained in a timely manner. Our work focuses on methods that are scalable to large networks, computationally efficient, and aim

to minimize the lag time from when the correlated events occur to when they are detected.

2.3.2 Discovering Functional Communities

In information networks, users broadcast messages to an audience of potential viewers, other network users who may choose to respond by broadcasting messages of their own, thus disseminating content across the network. A graph may be used to model the potential viewers of each broadcast, but this relationship does not imply information transfer. In fact, the exact paths through which information flows may not be directly observable. In Chapter 4 we explore how the event-driven network model can be used to study the diffusion of information even when explicit paths of information transfer are not known.

In particular, we look at group behavior in diffusion processes. While hypothetically each piece of information, or *meme*, may follow a completely different diffusion pattern, in real-world information networks certain paths appear more frequently than others. Therefore, one might hope to cluster network users based on the content of their broadcasts, such that users in the same cluster participate in the same memes, and users in different clusters participate in different memes. This, however, is not realistic either, since users may have multiple interests, and would be apt to participate in memes relevant to any of those interests. In this case, a soft clustering may be more appropriate, where each user can be a member of multiple clusters.

We take this one step further. A clustering, whether hard or soft, indicates which users are similar, but does not attribute that similarity to participation in particular memes. To achieve more detailed attribution, we propose *co-clustering* the users and memes simultaneously. The result is a set of *biclusters*, user-meme cluster pairs; a dense bicluster indicates a set of users with a high degree of participation in a specific set of memes. This more nuanced analysis of the behavioral correlations between users can lead to better recommendations of interesting memes, improved predictions of future user activity, and a better understanding of diffusion processes in information networks.

Our co-clustering methodology first defines a metric to evaluate the quality of a co-clustering, and then searches for a co-clustering that maximizes the value of the metric. In full generality, with no constraints on the metric function, this problem is NP-hard because the number of possible co-clusterings is exponential in the number of users and memes, and any one of them

could maximize the metric. In practice, it is therefore necessary to employ heuristics for the results to be useful. We propose an efficient heuristic algorithm to find a good co-clustering according to a given metric, as well as a class of metrics with properties that can be leveraged for further improvements in efficiency.

2.3.3 Modeling Collaboration in Academia

Successful research is often the result of collaborative efforts. In academia, this is manifested in individual researchers working together to publish joint papers. Graphs can be used to represent such collaboration networks, with an edge signifying a coauthorship relationship, optionally weighted by the number of joint papers. Alternatively, academic collaboration can be modeled as an event-driven network, where the publication of a paper is represented as an event with the coauthors as sources and the entire set of researchers as recipients.

In Chapter 5, we study the collaborative behavior of researchers in academia using tools from Game Theory. We define a game where each researcher is trying to maximize her own academic success, and use simulations to analyze the effects of this behavior on the individual researchers as well as the community as a whole. The flexibility of the event-driven network model helps us to study how collaboration strategies change over time.

We posit that due to the collaborative nature of academic research, an individual's success is typically not solely the result of his own efforts. Our model attempts to capture the way that an individual's success can be partially attributed to the contributions of his coauthors.

Our theoretical model makes certain assumptions about the academic world – for example, that each researcher considers the publication records of all other researchers, and that all pairs of researchers have equal opportunity to collaborate – that are not entirely realistic. However, the analytical results we derive under our theoretical model may indicate general principles that still apply under more realistic computational settings.

Chapter 3

Detecting Correlated Events in Communication Networks

3.1 Introduction

3.1.1 Background and Motivation

A naive model of event-driven networks consists of a set of entities whose behaviors and interactions are governed by independent processes. However, entities in real-world networks often exhibit correlated behavior, for example due to influence, environmental effects, or response to common external stimuli. These dependencies are usually not explicitly known. The ability to efficiently detect correlated events in large networks could therefore assist in a variety of application domains, such as computer network security, intelligence, marketing, knowledge discovery, and recommendation systems. For example, correlated activity in computer network traffic may indicate a coordinated attack or the spread of a virus. In this work we focus on communication networks, in which individual people or computers communicate with one another directly through channels such as email, IP connections, or face-to-face encounters.

A common approach to analyzing event-driven network data is to first aggregate information over fixed or variable-length time blocks. This approach facilitates the use of many existing analytical tools, but it imposes a trade-off: shorter time blocks give higher resolution at an increased storage cost and may introduce data sparsity issues; longer time blocks may have a smoothing effect that hides shorter-term deviations in behavior. Furthermore, different time granularities may be appropriate for different nodes or edges within the same network. In that case, choosing a global block size may bias analysis towards certain entities and overlook others, a phenomenon we call *time-scale bias*.

Sulo et al. propose a method for choosing the temporal resolution that best balances the trade-off between minimizing variance and minimizing information loss for a specific graph

metric [90]. Tong et al. [95] and Lian et al. [67] independently suggest multi-resolution approaches. Sharan et al. determine edge weights for a summary graph based on a decay model with global parameters [89]. These are all reasonable approaches to temporal analysis of event-driven networks, but as they all involve the discretization of time or depend on a global time parameter, they are all susceptible to time-scale bias.

In addition, many existing algorithms for network analysis are designed to be performed off-line. In the off-line scenario, the entire dataset is available simultaneously, and multiple passes over the data are permitted. When new data becomes available, it may require re-analyzing the entire dataset. In a real-world setting, where new data arrives continuously and at high rates, off-line approaches may not be able to provide up-to-date results in a timely manner. We use the term *streaming* to refer to models or algorithms that require only a single pass over a dataset in chronological order, typically with space and time constraints. In our case of event-driven networks, we aim for space requirements that are linear in the number of node pairs that ever communicate, regardless of the period of time over which data is collected.

These challenges motivate the need for new approaches to model temporal dynamics, overcome time-scale bias, and address efficiency concerns for analysis of event-driven networks.

3.1.2 Related Work

One popular approach to event-driven network analysis begins by constructing a (weighted or unweighted) graph to represent network activity aggregated over time, called a *summary graph*. Then, static graph algorithms employing techniques such as clustering, spectral analysis, and centrality analysis are applied. One application is anomaly detection, which looks for nodes or substructures that are statistical outliers in the graph based on some pre-defined measure. Noble et al. identify subgraphs that appear infrequently using a data mining tool called Subdue and a variant of the Minimum Description Length Principle [75]. Sun et al. identify outlier nodes in bipartite graphs based on properties of their neighborhood [92]. Akoglu et al. propose OddBall, which takes a similar approach for finding outlier nodes in a weighted summary graph [5].

A second approach begins by segmenting time into blocks and constructing a summary graph for each time block. Different nodes can then be compared based on their history of past communication and local graph structure. Priebe et al. represent the network as a sequence of

summary graphs, examining properties of each node's neighborhood over time [79]. Candia et al. study anomaly detection in spatio-temporal phone data, analyzing daily call volume from each cell tower and comparing to the mean call volume for that tower [13]. They use ideas from percolation theory to identify times and spatial regions of high activity.

A third approach, *change detection*, looks at how summary graphs change over time. Sun et al. propose the GraphScope algorithm, which clusters vertices in order to minimize the number of bits required to represent the graph, and a *change point* occurs when it would be more space-efficient to encode a summary graph by itself than in conjunction with the preceeding ones [91]. Henderson et al. measure properties of each summary graph, performing detailed analysis of community structure and individual node behavior only when there are significant changes in global metrics [41].

A fourth approach models communication data as a time series, and uses tools from signal processing to analyze patterns of communication for nodes, edges, or the whole network. Ihler et al. use a hidden Markov model to understand temporal patterns in network traffic volume and distinguish between normal and abnormal behavior [45]. Cao et al. learn a B-spline model, identifying both short-term deviations and long-term trends [14]. Lakhina et al. do statistical outlier detection on the time series of traffic volume across origin-destination flows, in terms of # of bytes, # of packets, and # of IP-flows [60]. Their approach is to use PCA to find the most prevalent trends across all flows (top-k eigenflows), and then mark a flow as anomalous at a particular time based on how well it matches the eigenflow prediction. Abello et al. also use a time series model, comparing the activity of each node or edge to the overall network behavior [3].

A related problem is that of asynchronous pattern matching in time series. Sakoe et al. introduce Dynamic Time Warping, which finds matching patterns between a pair of discrete time series, even if one of the time series is stretched or compressed along the temporal dimension or has missing information [87]. Li et al. propose Parsimonious Linear Fingerprinting, which indentifies similar pairs of continuous time series, allowing for changes in amplitude, frequency, and phase shifting [66].

Another approach to dynamic network analysis, the *Temporal Path Model*, considers paths composed of time-ordered edge events. Xie et al. propose an approach for identifying the origin

of viruses in a network using *moonwalks*, random walks on the graph backward in time [98].

All of the above network analysis approaches involve the discretization of time or use a model with global time parameters, and are thus susceptible to time-scale bias. In addition, many of them are off-line algorithms, making them impractical for real-time analysis of streaming data from communication networks. Our approach, based on techniques from Renewal Theory, is a first step towards addressing these challenges.

3.1.3 Contributions and Outline

We first present a stochastic network model based on ideas from Renewal Theory. Through experiments we demonstrate that this is a reasonable model and apply it to detect correlated events in real-world communication networks.

Our contributions can be summarized as follows:

- The REWARDS model, a streaming stochastic model for systems of point processes
- A formal definition of recency for renewal processes that is time scale-invariant
- A statistical method for measuring correlation in event-driven networks that addresses the variety of temporal characteristics present in real-world networks
- A streaming local algorithm for detecting correlated activity among a fixed set of nodes
- An efficient global algorithm that simultaneously detects subsets of nodes exhibiting correlated activity in disparate parts of the network

We begin by introducing our stochastic model, the REWARDS model, in Section 3.2.1. In Sections 3.2.2 and 3.2.3 we propose statistical methods to measure dependencies in a network using the REWARDS model. We present two algorithms for detecting correlated events in Section 3.2.4, and analyze their running times in Section 3.2.5. Experimental results are provided in Section 3.3. Section 3.4 provides further discussion of the strengths and limitations of our approach, the significance of our work, and directions for future work.

3.2 Methodology

Our approach is to model an event-driven network as a system of stochastic processes. Given data from a real-world network, we first learn parameters of the model from the data, and then

apply statistical tools to identify dependencies between processes.

3.2.1 The REWARDS Model

A *renewal process* Φ is a continuous-time Markov process in which states correspond to the natural numbers, and state transitions $i-1 \rightarrow i$ occur with waiting times $w_i \in \Re^+$ sampled independently from the same distribution. This yields a sequence of times $t_i = \sum_{k=1}^{i} w_k$ at which state transitions occur. We refer to the countable ordered set $T_\Phi = \{t_i\}$ as the *time sequence* for Φ.

Consider a communication network where data arrives as a stream of time-stamped messages sent directly from one node in the network to another. We represent this as an event-driven network $\mathcal{N} = (\mathcal{U}, \mathcal{E})$, with events corresponding to the messages. For each pair of nodes (u, u') that ever interacts, we extract the discrete-event sequence $\mathcal{E}_{(u,u')}$ consisting of the relevant events, as well as the corresponding time sequence $T_{(u,u')}$ (see Section 2.1.2).

The REWARDS (REneWal theory Approach for Real-time Data Streams) model represents the network as a system of renewal processes. For each (directed or undirected) node pair (u, u'), we infer the waiting time distribution for the renewal process $\Phi_{(u,u')}$ that generated the time sequence $T_{(u,u')}$.[1] The choice of distribution model and inference method are independent of our work.

Now that we have introduced the REWARDS model, we present a statistical methodology that will help in analyzing dependencies between processes.

3.2.2 Measuring Statistical Correlation

We suggest the following procedure to test for positive correlation of random variables. Given a set of samples $x_1, \ldots, x_n$ taken from real-valued continuous random variables $X_1, \ldots, X_n$:

1. Let Y_i be a normalization of X_i, i.e. $Y_i \sim \textit{Uniform}(0, 1)$, for all $1 \leq i \leq n$
2. Apply a statistical goodness-of-fit test under the null hypothesis that the corresponding values $y_1, \ldots, y_n$ are i.i.d. samples from $\textit{Uniform}(0, 1)$

[1] The REWARDS approach could also be used to model individual nodes or other network substructures. In this work, we focus on applications where the mode of communication is one-to-one messages.

3. Measure likelihood of the sample as the p-value from the goodness-of-fit test

Step 1 is accomplished using a technique called the *probability integral transform*: Let X be a continuous random variable, and let F_X be the corresponding cumulative distribution function (CDF). Let Y be the random variable defined by applying F_X to X, i.e. $Y = F_X(X)$.

Proposition 3.1. $Y \sim$ *Uniform*$(0, 1)$.

Proof.

$$\begin{aligned} F_Y(y) = Pr(Y \leq y) &= Pr\left(F_X(X) \leq y\right) && \text{[by definition]} \\ &= Pr\left(F_X^{-1}\left(F_X(X)\right) \leq F_X^{-1}(y)\right) && [F_X^{-1} \text{ increasing}] \\ &= Pr\left(X \leq F_X^{-1}(y)\right) = F_X\left(F_X^{-1}(y)\right) = y && \text{[by definition]} \end{aligned}$$

□

The probability integral transform is illustrated in Figure 3.1. Values of X are along the x-axis, while Y takes values in $[0, 1]$, shown along the y-axis.

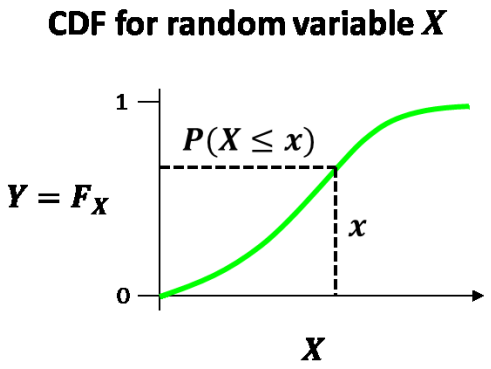


Figure 3.1: The probability integral transform

For Step 2, we perform a statistical goodness-of-fit test comparing the empirical distribution function $\widehat{Y}(y) = \frac{1}{n} \cdot |\{i : y_i \leq y\}|$ to the theoretical cumulative distribution function for *Uniform*$(0, 1)$ samples, *Triangle*$(0, 1)$. One such test, the Kolmogorov-Smirnov test [57], is illustrated in Figure 3.2. The Kolmogorov-Smirnov statistic, denoted d_{KS}, is the maximum

difference between the two distributions:[2]

$$d_{KS}(\widehat{Y} \parallel \textit{Triangle}(0,1)) = \max_y(\widehat{Y}(y) - y).$$

The p-value, denoted p_{KS}, is the likelihood that a sample generated according to the null model would yield a difference of at least d_{KS}. The smaller the p-value, the greater the confidence with which one may reject the null hypothesis, and therefore the stronger the implication that the random variables are not independent.

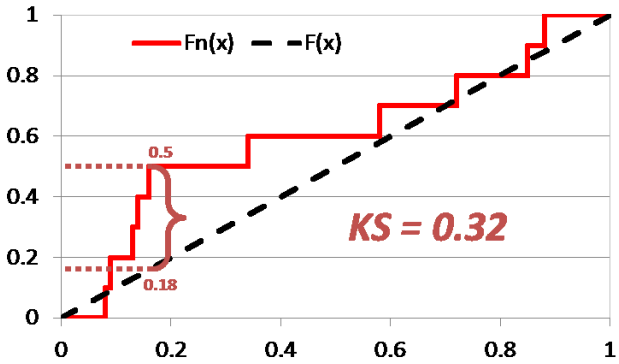


Figure 3.2: The Kolmogorov-Smirnov statistic

Among probability distance functions, we choose the Kolmogorov-Smirnov statistic because of its interpretability: it indicates the threshold $\gamma_{KS} = \arg\max_y(\widehat{Y}(y) - y)$ with the highest concentration of values falling below that threshold, as well as the corresponding values. This feature is instrumental for attribution, allowing us to identify the variables with the highest correlation. Next, we explore how this paradigm can be applied to detect correlated events in event-driven networks.

3.2.3 Recency

Our goal is to detect correlations in communication activity. In particular, we look for times at which there has been an unusually high concentration of recent activity amongst a subset of nodes in the network. In order to formalize this task, we first define what we mean by *recent*.

[2]In our experiments we use the positive one-sided Kolmogorov-Smirnov statistic since for the applications we consider, we are more interested in correlation of activity than non-activity. A different choice may be appropriate for other applications, such as fault detection.

A natural quantity to consider is the *age* of a renewal process, which is defined in the Renewal Theory literature as the time elapsed since the last event:

$$\mathrm{Age}_\Phi(t) = \begin{cases} t - t_i & \text{if } t_i \le t \text{ and } \nexists\, t_j : t_i < t_j \le t \\ \infty & \text{if } \nexists\, t_i \le t \end{cases}$$

However, activity rates may vary among nodes in the network, and looking at the age will bias analysis towards nodes with higher activity rates, potentially hiding correlated behavior among lower-rate nodes. To compensate for this time scale bias, we first normalize each process following the procedure proposed in Section 3.2.2.

Let F_Φ^{Age} denote the limit distribution of the Age function:[3]

$$F_\Phi^{\mathrm{Age}}(\tau) = \lim_{t\to\infty} Pr\left(\mathrm{Age}_\Phi(t) \le \tau\right).$$

We define the *recency* of Φ at time t to be

$$\mathrm{Rec}_\Phi(t) = 1 - F_\Phi^{\mathrm{Age}}\left(\mathrm{Age}_\Phi(t)\right).^{4}$$

Note that Rec is a decreasing function on every interval $[t_i, t_{i+1})$ (see Figure 3.3), and that it satisfies the uniformity property as described in Section 3.2.2, that $Rec_\Phi \sim$ *Uniform*$(0,1)$ for any renewal process Φ. The intuition is that sampling the recency function randomly in time will generate uniform random samples in $[0,1]$. This normalization is scale-invariant (recency values remain the same when time is stretched by a constant factor), which makes our approach robust to differences in time scale between networks or between entities within the same network.

Next, we consider correlation of recent activity across multiple processes. Given a set Ω of renewal processes, we define the recency of Ω at time t as $\mathrm{Rec}_\Omega(t) = 1 - p_{KS}$, where p_{KS} is the p-value from performing the Kolmogorov-Smirnov test as described in Section 3.2.2. Larger values of Rec_Ω are a stronger indication that the processes are not independent.

For notational convenience, we use $\mathrm{Age}_{(u,u')}(t)$ to denote the age of the renewal process corresponding to (u,u'), $\mathrm{Rec}_{(u,u')}(t)$ to denote the recency of the renewal process corresponding to (u,u'), and $\mathrm{Rec}_S(t)$ to denote the recency of the set of renewal processes corresponding

[3]Note that this is different from the distribution of waiting times for the renewal process. As an illustration of this, consider the inspection paradox.

[4]We define recency using $1 - CDF$ instead of CDF to match the linguistic intuition that higher recency corresponds to more recent activity.

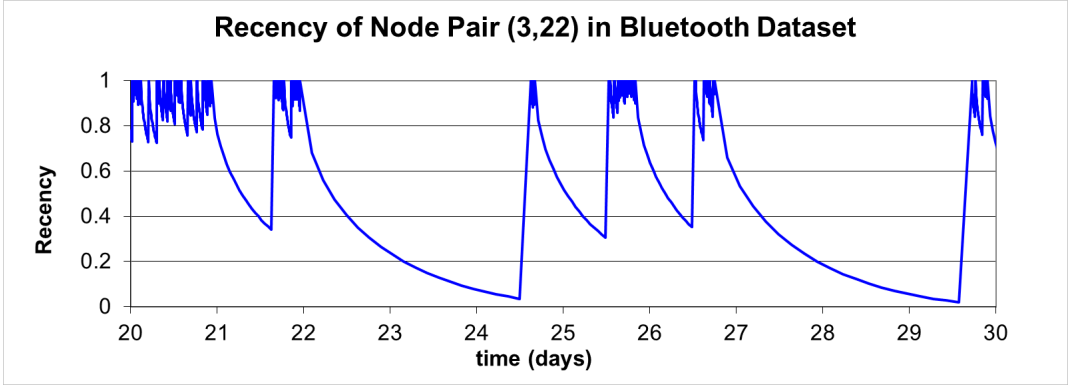


Figure 3.3: Plot of recency over time for an edge in the Bluetooth network. If the model is accurate, the value of the recency function will be $\leq \rho$ exactly ρ fraction of the time, for any $\rho \in [0, 1]$.

to node pairs in S.

3.2.4 Correlated Event Detection

With the infrastructure in place, we now present two efficient algorithms to detect recent correlated activity in communication networks. For both algorithms, we use the REWARDS approach to model activity for each node pair (u, u') as a renewal process $\Phi_{(u,u')}$, and then detect sets of correlated events using the statistical approach in Section 3.2.2.

The first is a streaming local algorithm, L-CORE (Local algorithm for detecting CORrelated Events), which simultaneously monitors the neighborhood of each node in the network. Whenever there is outgoing communication from a node, the recency of its neighborhood is computed, and a flag is thrown if it exceeds a pre-specified sensitivity threshold. Algorithm 3.1 gives a formal description of the L-CORE algorithm.

The second algorithm, G-CORE (Global algorithm for detecting CORrelated Events), is a static algorithm built on the streaming REWARDS model, which searches the entire network for the subsets of nodes with the highest concentrations of recent activity at a given time. This is accomplished by maintaining a disjoint set data structure on $\mathcal{U}$ and running a variant of the Union-Find Algorithm [94], incrementally considering node pairs (u, u') in decreasing order of recency. At each iteration, if u and u' are not already in the same set, then we join the two set into one. The G-CORE algorithm is given as Algorithm 3.2, and it is illustrated in Figure 3.4.

We note that a pair of disjoint sets is a refinement (subpartition) of the single set formed

Algorithm 3.1 The L-CORE Algorithm (Local algorithm for detecting CORrelated Events)

Input: An event-driven network $\mathcal{N}$ consisting of a set of nodes $\mathcal{U}$ and a stream of communication events $\mathcal{E}$, and a sensitivity threshold θ.

Output: For each time at which correlated activity is detected, the source node that is responsible for the activity along with the corresponding set of events.

1: For notational convenience, let $N^-(u) = N^-_{G_{\mathcal{E}_u}}(u)$; that is, the set of nodes $u' \in \mathcal{U}$ that have received communication from u. For each node $u \in \mathcal{U}$, maintain the (approximate) distribution of inter-arrival times for outgoing communication to each node $u' \in N^-(u)$.

2: Every time t there is outgoing communication from u:

- Update the inter-arrival time distribution for the corresponding recipient node(s).
- Compute the recency of all outgoing activity from u: $\mathrm{Rec}_{S_u}(t)$, where $S_u = \{(u, u') : u' \in N^-(u)\}$.
- If $\mathrm{Rec}_{S_u}(t) > 1 - \theta$, output the tuple $(u, t, \mathrm{Rec}_{S_u}(t), \gamma_{KS}, \mathcal{E}^*_u(t))$, where $\mathcal{E}^*_u(t)$ contains the most recent event from u for each $u' \in N^-(u)$ such that $\mathrm{Rec}_{(u,u')}(t) \leq \gamma_{KS}$.

Algorithm 3.2 The G-CORE Algorithm (Global algorithm for detecting CORrelated Events)

Input: An event-driven network $\mathcal{N} = (\mathcal{U}, \mathcal{E})$, with the IAT distributions for all node pairs that communicate through time t.

Output: A list of node sets partitioning $\mathcal{U}$, along with the recency values corresponding to their induced subgraphs.

1: Construct the graph $G = (\mathcal{U}, E)$ induced by the set of all events up to time t, each edge $(u, u') \in E$ weighted by the corresponding recency value $\mathrm{Rec}_{(u,u')}(t)$.

2: Initialize a disjoint set data structure on $\mathcal{U}$:

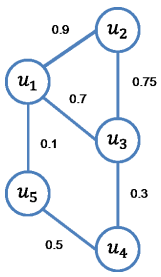
- Set Π, the current partition of $\mathcal{U}$, and Π^*, the output partition, to be the collection of singleton vertices: $\Pi = \Pi^* = \{\{u\} : u \in \mathcal{U}\}$.
- For each set $U \in \Pi$, maintain $\mathrm{Rec}^*_U(t)$, the highest recency attained by any of its subsets (including itself), which is initialized to 0.

3: Sort $E(G)$ in decreasing order of weight.

4: For each edge $(u, u') \in E(G)$, if u and u' are in different sets:

- Update Π by removing the sets containing u and u' and adding their union U.
- Compute $\mathrm{Rec}_U(t) = \mathrm{Rec}_S(t)$, where S is the set of edges incident to nodes in U, and update $\mathrm{Rec}^*_U(t)$ as necessary.
- If $\mathrm{Rec}_U(t)$ is greater than the recency of all subsets of U (i.e. $\mathrm{Rec}^*_U(t) = \mathrm{Rec}_U(t)$), update Π^* by adding U and removing its subsets.

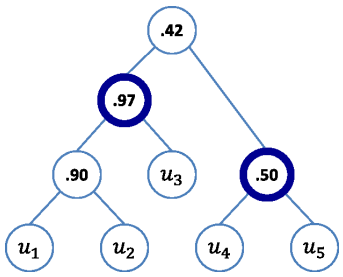
5: Output Π^* along with the corresponding values of $\mathrm{Rec}_U(t)$ for each $U \in \Pi^*$.



(a)

Edge	Wt.	Node Set U	Rec_U
(u_1, u_2)	0.9	$\{u_1, u_2\}$	0.900
(u_2, u_3)	0.75	$\{u_1, u_2, u_3\}$	0.973
(u_1, u_3)	0.7	$\{u_1, u_2, u_3\}$	-----
(u_4, u_5)	0.5	$\{u_4, u_5\}$	0.500
(u_3, u_4)	0.3	$\{u_1, u_2, u_3, u_4, u_5\}$	0.421
(u_1, u_5)	0.1	$\{u_1, u_2, u_3, u_4, u_5\}$	-----

(b)



(c)

Figure 3.4: (a) The weighted graph constructed in Step 1 of the G-CORE algorithm. (b) Iterations of the loop in Step 4. (c) The G-CORE tree, showing the recency of each node set, with the output partition Π^* highlighted.

by their union. We can therefore construct a partially-ordered set (poset) over sets considered during the G-CORE algorithm, ordered by the containment relation. This poset is in fact a tree, where the leaves are individual nodes and the root is $\mathcal{U}$.[5] An example of this hierarchical structure can be seen in Figure 3.4(c). We refer to this poset as the *G-CORE tree*.

Two elements a, b in a poset are *comparable* if $a \leq b$ or $b \leq a$. A *maximal antichain* of a poset is a maximal set of elements of the poset, no two of which are comparable. We note that any maximal antichain in the G-CORE tree forms a partition of $\mathcal{U}$. The output of the G-CORE algorithm is a list of event sets, ranked in decreasing order of recency. The graphs induced by these event sets are guaranteed to partition $\mathcal{U}$ since Π^* is always a maximal antichain in the G-CORE tree. (It is true initially, and is maintained every time Π^* is modified in Step 4.)

This property is instrumental to the G-CORE algorithm's ability to detect correlated activity in disparate parts of the network simultaneously. Without the disjointedness constraint, the node sets with the second and third highest correlated activity would likely be subsets or supersets of the top result. This is particularly relevant for real-world security applications, where an attacker may try to hide malicious activity by creating a diversion in another part of the network.

The G-CORE algorithm is an effective way to detect correlated activity in an event-driven network. However, because its worst-case running time makes it impractical for many real-world networks (see Section 3.2.5), we propose an efficient heuristic algorithm with the same

[5] If $G_{\mathcal{E}}$ is not connected, this holds for each connected component.

input and output specifications plus a precision parameter, given as Algorithm 3.3.

Algorithm 3.3 The Heuristic G-CORE Algorithm

Input: An event-driven network $\mathcal{N} = (\mathcal{U}, \mathcal{E})$, with the IAT distributions for all node pairs that communicate through time t, and a precision parameter $0 < \delta < 1$.

Output: A list of node sets partitioning $\mathcal{U}$, along with the recency values corresponding to their induced subgraphs.

1: Construct the graph $G = (\mathcal{U}, E)$ induced by the set of all events up to time t, each edge $(u, u') \in E(G)$ weighted by the corresponding recency value $\mathrm{Rec}_{(u,u')}(t)$.

2: Initialize a disjoint set data structure on $\mathcal{U}$:

- Set Π, the current partition of $\mathcal{U}$, and Π^*, the output partition, to be the collection of singleton vertices: $\Pi = \Pi^* = \{\{u\} : u \in \mathcal{U}\}$.
- For each set $U \in \Pi$, maintain $\mathrm{Rec}^*_U(t)$, the highest recency attained by any of its subsets (including itself), which is initialized to 0.

3: Partition the interval $[0, 1]$ into equally-sized subintervals of width δ, and bin each edge according to the subinterval containing its recency value.

4: For each subinterval, in increasing order:

- Add all edges in the corresponding bin, and update the disjoint sets accordingly.
- Compute the recency $\mathrm{Rec}_U(t)$ of each set $U \in \Pi$, and update $\mathrm{Rec}^*_U(t)$ as necessary.
- For each set $U \in \Pi$, if $\mathrm{Rec}_U(t)$ is greater than the recency of all subsets of U (i.e. $\mathrm{Rec}^*_U(t) = \mathrm{Rec}_U(t)$), update Π^* by adding U and removing its subsets.

5: Output Π^* along with the corresponding values of $\mathrm{Rec}_U(t)$ for each $U \in \Pi^*$.

3.2.5 Complexity Analysis

Both the local and global algorithms are based on the REWARDS approach, which models each node pair as being generated by a renewal process, and infers the waiting time distribution from the observed inter-arrival times (IATs) between events. Streaming algorithms to dynamically maintain the (approximate) IAT distribution for each node pair (e.g. using histograms [9, 38], kernel density estimators [84, 59, 76], or other methods [24, 10]) enable the use of the REWARDS model for efficient analysis of streaming network data. The choice of distribution model and approximation method are independent of our work.

In our experiments, we use maximum-likelihood estimation to determine the best-fit parameters for a Bounded Pareto distribution (see Section 3.3.1). The Bounded Pareto has three parameters: x_{min}, the minimum possible IAT; x_{max}, the maximum possible IAT; and α, the shape parameter. The maximum-likelihood estimator for these parameters can be determined

using only the following three values: the minimum IAT seen so far, the maximum IAT seen so far, and the sum of the logs of all IATs seen so far [1]. Since these values can be maintained and updated in a streaming fashion, only $O(1)$ space is required to model each time sequence, with constant-time update per communication event.

In the L-CORE algorithm, each node u maintains the IAT distributions for outgoing activity to its neighbors $N^-(u)$ (Step 1). For each outgoing event, we update the corresponding IAT distribution, calculate the recency of communication with each neighbor,[6] and then compute the collective recency for all outgoing activity (Step 2). Each of these steps is at most linear in the number of neighbors. Therefore, the running time for the L-CORE algorithm at node u is $O(|N^-(u)|)$ for each outgoing communication event from u.

The G-CORE algorithm takes as input the IAT distributions for all node pairs in the network that communicate, updated through time t. Let $n = |\mathcal{U}|$, and let m be the number of such node pairs. In Step 1, a weighted graph G is constructed by computing the recency of each node pair, which takes $O(n+m)$ time. Initializing the disjoint set data structure takes $O(n)$ time (Step 2). Sorting the edges by weight takes $O(m \log m)$ time (Step 3). For each of the $n-1$ iterations of the loop in Step 4 when the union operation is performed, we compute the recency of a subset of nodes, which runs in time linear in the number of incident edges, which can be at most m, for a total of $O(n \cdot m)$ time for the loop. The complexity of maintaining the disjoint set data structure is $O((n+m) \cdot \alpha(m,n))$ over the course of the algorithm using path compression and union by rank, where $\alpha(m,n)$ is the extremely slowly-growing inverse Ackermann function, which is a small constant for all practical values of n and m [94]. Finally, it takes $O(n+m)$ time to output the sets in the partition along with the corresponding edges and recency values (Step 5). Thus the worst-case running time of the G-CORE algorithm is $O(n \cdot m)$.

For the Heuristic G-CORE algorithm, Steps 1 and 2 still take $O(n+m)$ and $O(n)$ time, respectively. Binning the edges into subintervals by weight takes $O(m)$ time (Step 3). For each of the $\frac{1}{\delta}$ iterations of the loop in Step 4, we compute the recency of all sets in Π, which takes $O(m)$ time since each edge is incident to at most two node sets, for a total of $O\left(m \cdot \frac{1}{\delta}\right)$ time for the loop. The complexity of maintaining the disjoint set data structure is still $O((n+$

[6] For the Bounded Pareto model, there is no closed formula for the limit distribution F^{Age}, so in our implementation we use Simpson's Method for numerical integration.

Dataset	Description	Timespan	# of nodes	# of edges	# of events
ENRON [55]	email	2 years	1141	2017	4847
BLUETOOTH [28]	physical proximity	9 months	101	2815	102563
LBNL [77]	IP traffic	1 hour	3317	9637	9258309
TWITTER [20]	@ replies	1 year	262932	307816	1134722

Table 3.1: Datasets used in our experiments

$m) \cdot \alpha(m, n))$. Finally, it takes $O(n + m)$ time to output the sets in the partition along with the corresponding edges and recency values (Step 5). Thus the worst-case running time of the Heuristic G-CORE algorithm is $O\left((n + m) \cdot \alpha(m, n) + m \cdot \frac{1}{\delta}\right)$.

3.3 Evaluation

To evaluate our approach, we perform experiments on several simulated and real-world networks. Table 3.1 lists the real-world datasets used in our experiments.

3.3.1 Modeling Inter-Arrival Times

To apply the REWARDS model, we must estimate the distribution of inter-arrival times across each network edge. [11] suggests that inter-arrival times for communication follow a power law. In Figure 3.5, we plot the probability mass function (PMF) of inter-arrival times for several network edges in each of our datasets using a logarithmic binning procedure. The linearity of the distributions (on a log-log scale) indicates that this claim holds across all of our datasets, regardless of the communication medium. An interesting phenomenon is seen in the LBNL packet trace data, which appears to be the sum of two power-law distributions. We hypothesize that two processes are being observed, one corresponding to inter-arrival times of consecutive packets in a single connection, and the other corresponding to inter-arrival times between connections.

Next, we consider several variants of the Pareto distribution, a power-law distribution commonly used for modeling real-world phenomena [74]. The Pareto distribution is parameterized by two variables: x_{min}, the smallest value with non-zero probability density, and α, which measures the peakedness of the distribution. The Bounded (or Truncated) Pareto has an additional parameter: x_{max}, the largest value with non-zero probability density.

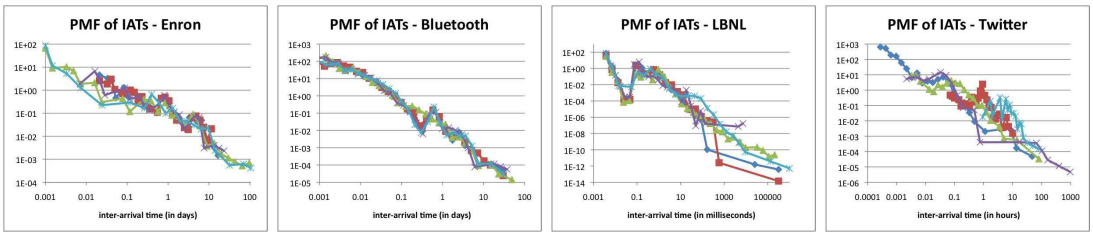


Figure 3.5: IAT distributions for the 5 most active edges in each of our datasets on a log-log scale. Linearity indicates that they obey a power law. IATs between packets and between connections may account for the bimodal distribution in the LBNL dataset.

We compared the Pareto, Bounded Pareto, and Exponential distributions for modeling the distribution of inter-arrival times along edges in the four datasets, using both the Maximum-Likelihood Estimation and Mean Estimation techniques to estimate the distribution parameters, and found that the Bounded Pareto Distribution using a Maximum-Likelihood approach consistently out-performed the other models. Therefore, we use the Bounded Pareto with Maximum-Likelihood Estimation for all further experiments.

Using an upper-bounded model makes sense intuitively, since there are frequently practical limits on the maximum time gap between consecutive communication along an edge. For example, if there has been no communication between two people in 100 years, it may be reasonable to assume that communication has ceased permanently. Therefore, in practice, if the time elapsed since the last communication is greater than the estimated maximum value, we designate the edge as *dead* and omit it in computation. If, however, communication is observed along an edge previously considered dead, we conclude that the estimated maximum value was inaccurate, and include the edge in further computation with updated parameter estimates.

3.3.2 Robustness to Time Scale

To evaluate the ability of the REWARDS approach to detect correlated activity regardless of time scale, we perform a series of experiments on simulated networks. First, we outline a general procedure for simulating a communication network:

1. Generate a graph to represent the nodes and underlying relationships in the network.
2. Choose values for parameters α and β.

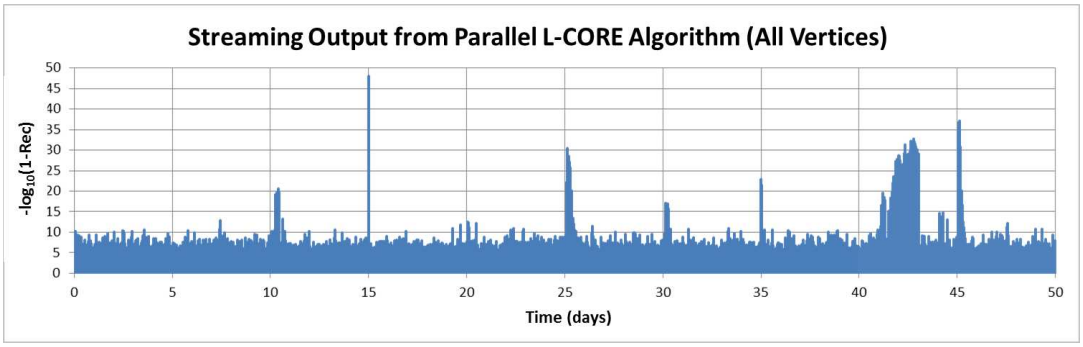
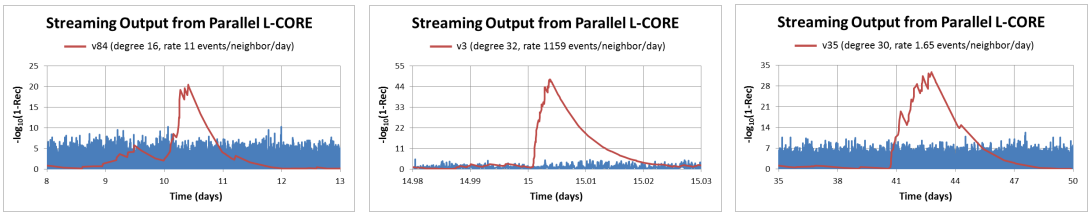


Figure 3.6: Output from running the L-CORE algorithm on a simulated network, underlying graph generated by the R-MAT model with 128 vertices and average degree 16, IATs for edge activity sampled from Bounded Pareto distributions with $\alpha = 1$ and varying scale parameters.



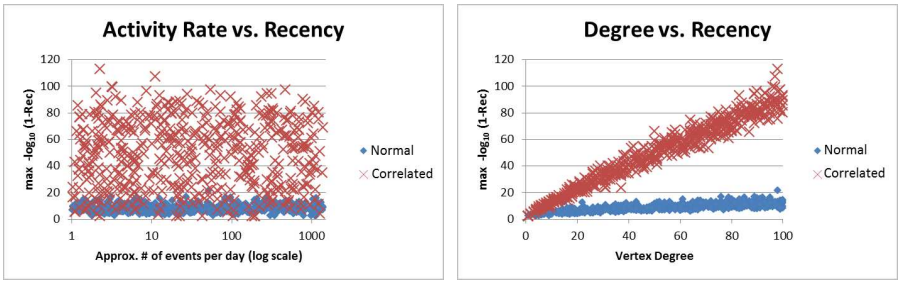
(a) (b) (c)

Figure 3.7: Same as in Figure 3.6, but zoomed in to highlight correlated activity at times (a) $t = 10$, (b) $t = 15$, and (c) $t = 40$.

3. For each edge (u, u'), select a rate parameter $r_{(u,u')}$ indicating the frequency of activity, and generate a discrete-time sequence $T_{(u,u')}$ by simulating a renewal process with inter-arrival times sampled from a Bounded Pareto distribution with shape parameter α, minimum inter-arrival time $1/r_{(u,u')}$, and maximum inter-arrival time $c \cdot 1/r_{(u,u')}$.

4. To simulate correlated activity across a set of edges S at time t, increase the rate parameter for each of the edges in S by multiplicative factor β.

We use the RMAT model proposed in [17] to generate a graph with power-law degree distribution and small world characteristic. Then we select a rate parameter r_u for each node u sampled randomly between once a week and once every ten minutes, and simulate communication along the outgoing edges from u with rate parameter $r_{(u,u')} = r_u$. Every five days, we randomly select a node u to simulate correlated activity at 10x the usual rate for a duration of five times the new minimum inter-arrival time. Figure 3.6 plots the output from the L-CORE algorithm, the recency of outgoing activity from each node in the simulated network over time.



(a) (b)

Figure 3.8: Correlation between (a) activity rate and recency, and between (b) size of neighborhood and recency, over 1000 randomized trials.

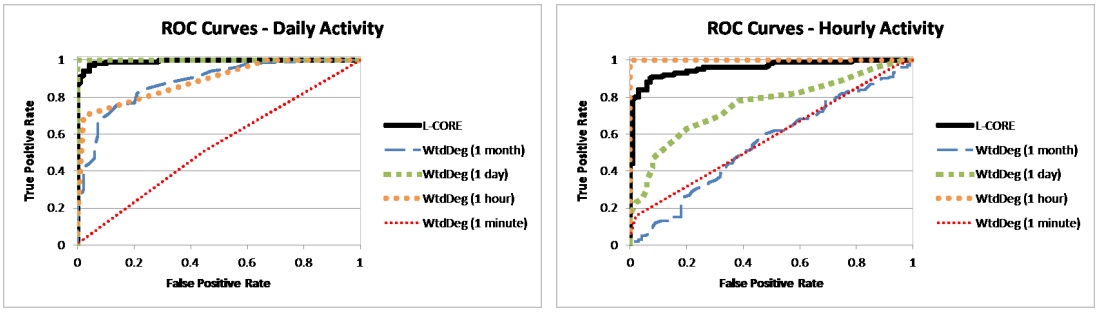
That is, if node u has outgoing activity at time t, there is a point $(t, \mathrm{Rec}_u(t))$.[7]

We observe a clear peak during seven of the nine 5-day periods. Inspection of the data confirms that each of those peaks indeed corresponds to the node with simulated correlated activity during that period. However, some peaks are higher than others, and some are very sharp compared to others which appear to be more gradual. In Figure 3.7 we examine several of these peaks more closely, and find that they in fact have similar shape up to horizontal and vertical scaling.

While it takes longer for correlated activity to be recognized at nodes with lower activity rates, we observe that the *magnitude* of the peak seems to be indepedent of the frequency of communication. We test this observation by performing an experiment with 1000 randomized trials. For each trial, we simulate a star network, randomly choosing the number of outgoing edges from the center node u as well as the activity rate r_u. In half of the trials, we also add correlated activity at 10x the normal rate. Figure 3.8(a) plots the activity rate r_u against $\max_t \mathrm{Rec}_u(t)$ for each trial. The Pearson correlation coefficient is 0.007 for the normal activity and -0.019 for the correlated activity, indicating that there is no significant correlation. This is consistent with our claim that *recency is time scale invariant*.

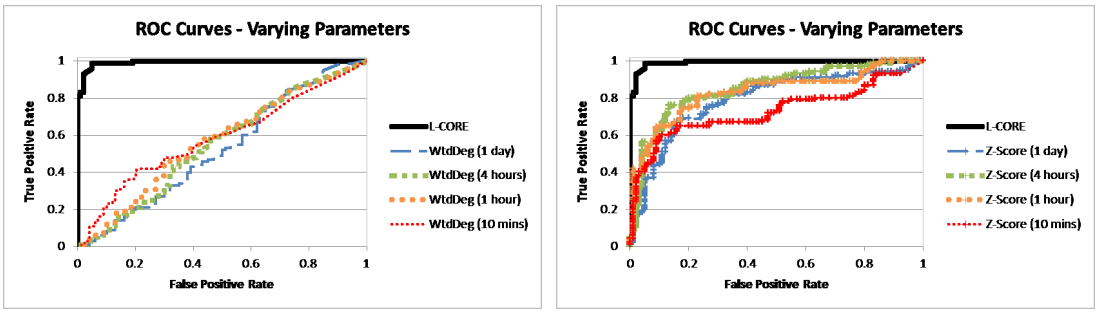
Next we analyze the correlation between $\max_t \mathrm{Rec}_u(t)$ and the size of the neighborhood $N^-(u)$. Since recency measures the unlikelihood that a burst of recent activity across a set of edges would occur by chance, we expect that a greater number of edges with increased activity

[7] For clarity of visualization, all plots use $-\log_{10}(1 - \mathrm{Rec})$, which scales the values but preserves their relative ordering.



(a) (b)

Figure 3.9: ROC curves showing the trade-off between accuracy and precision for several correlation detection methods, evaluated on simulated networks where each node has 10 outgoing neighbors, and (a) daily and (b) hourly normal activity rates. The z-score metrics perform almost identically to the corresponding weighted degree metrics, so are not shown.



(a) (b)

Figure 3.10: ROC curves showing the trade-off between accuracy and precision for several correlation detection methods, evaluated on simulated networks with randomized degree distributions and activity rates. Our approach out-performs both the (a) weighted degree and (b) z-score metrics.

would yield higher recency. Using the same set of 1000 trials as above, we get a Pearson correlation coefficient of 0.777 for normal activity and 0.980 for correlated activity. Note that in this experiment a larger neighborhood yields more communication events in total, each resulting in another recency computation. This sampling bias may account for the moderate positive correlation in normal activity. This effect is compounded by the presence of correlated activity, as evidenced by the much stronger Pearson correlation and the clear increasing trend in Figure 3.8(b).

3.3.3 Accuracy and Precision

We compare the accuracy and precision of the REWARDS approach in distinguishing between normal and correlated behavior with several baseline approaches and related work. Given time sequences corresponding to outgoing activity from a single source node, we consider several methods for detecting correlations in recent activity.

The first approach is based on the number of edges that have been active over a designated period of time: the more active edges, the higher the level of recent activity. This metric is equivalent to the degree of the node in the corresponding summary graph. The second approach is based on the total number of communication events that have occurred during a period of time, and is equivalent to the weighted degree of the node in the summary graph. The third and fourth approaches are based on [79], which suggests comparing the activity of a node or subgraph to its past behavior. Instead of using the raw value of a node property such as degree or weighted degree, they compute the z-score of the value at time t using the sample mean and standard deviation of the values at times $1, \ldots, t-1$.

All of the four approaches require segmenting time into blocks to create summary graphs, so we perform each of our experiments using a wide range of time block sizes. We found that the weighted degree-based metrics consistently out-perform the degree-based metrics on all our experiments, so we omit the degree-based metrics from the results.

In the first experiment, we simulate a network of 200 nodes, each with 10 outgoing edges and normal activity rate of one event per day, for a period of one hundred days. For half of the nodes we simulate normal activity, and for the other half we include include 12 consecutive hours of "correlated" activity at 10x the normal rate. We then consider the ability of several correlation detection methods to distinguish between nodes with normal and correlated behavior. The ROC curve visualizes the trade-off between precision and accuracy. Each point on the ROC curve indicates the false positive and true positive rates for one possible threshold value. Note that since in this experiment all nodes have the same degree and activity rate, the z-score metrics are essentially scaled versions of the simpler degree-based and weighted degree-based metrics, so are not shown.

Figure 3.9(a) compares the ROC curve for the L-CORE algorithm with those for the weighted

degree approach. We observe that the weighted degree-based metric with 1-day time blocks performs exceedingly well here, since it is specifically tailored to detect correlated activity that occurs on a daily time scale, followed closely by our REWARDS-based approach. The 1-month and 1-hour time blocks yield moderately good results, whereas the 1-minute time blocks are too fine-grained to capture correlations at the daily time scale.

Figure 3.9(b) shows results for a similar experiment, except with a normal activity rate of one event per hour, and correlated activity at 10x that rate for a period of half an hour. Here we observe that the weighted degree-based metric with 1-hour time blocks achieves perfect accuracy and precision, with L-CORE falling in a close second place. The method based on 1-day time blocks, which performed the best in the previous experiment, yields only mediocre results; 1-month time blocks are too coarse to detect correlated activity at an hourly rate; and 1-minute time blocks are again too fine-grained.

Next we move on to a more realistic setting, where nodes may have different numbers of outgoing edges and activity rates. For each trial, we select the degree of the node randomly between 10 and 100, the normal activity rate between once a minute and once every 30 days (using a logarithmic distribution to encourage sampling from the full range of time scales), and correlated activity between 5x and 10x the rate of normal activity. We compare with both the raw weighted degree metrics and the weighted degree z-score metrics in Figure 3.10. Since the degree and activity rate of the nodes can vary, the raw weighted degree metrics are highly skewed towards nodes of high degree and high activity rate, which is reflected in the results, performing not much better than random chance. The z-score metrics perform better, but their high accuracy and precision for specific time scales is compromised by their poor performance at others. The time scale invariance property of our REWARDS-based approach makes it *robust to variations in temporal dynamics*, yielding high performance across the board.

3.3.4 Detection Latency

One important goal in real-world correlated event detection is minimizing the time from when correlated activity occurs to when it is detected. We compare the detection latency of the L-CORE algorithm with that of the GraphScope algorithm, which creates summary graphs and then detects times at which there is significant change in the graph structure [91].

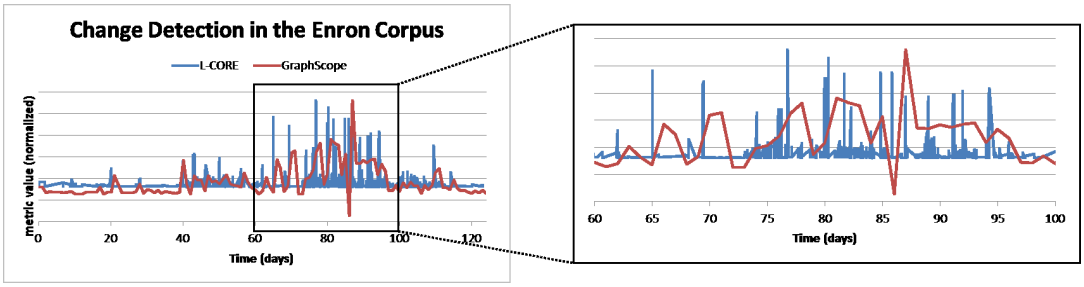


Figure 3.11: The L-CORE algorithm vs. the GraphScope algorithm (using 1-week time blocks as in [91]) for the ENRON dataset.

Figure 3.11 shows the results of both L-CORE and GraphScope on the ENRON dataset. First, we notice that peaks in the L-CORE output largely coincide with higher values from GraphScope, indicating that the two methods are identifying similar changes in network behavior. Furthermore, the peaks in the L-CORE output consistently *precede* the corresponding spikes in the GraphScope results. The time segmentation approach employed in [91] means that up to a week may pass before network changes are reflected in the analysis, whereas our approach based on the REWARDS model can detect changes as soon as they occur.

3.3.5 Scanning Activity in IP Traffic

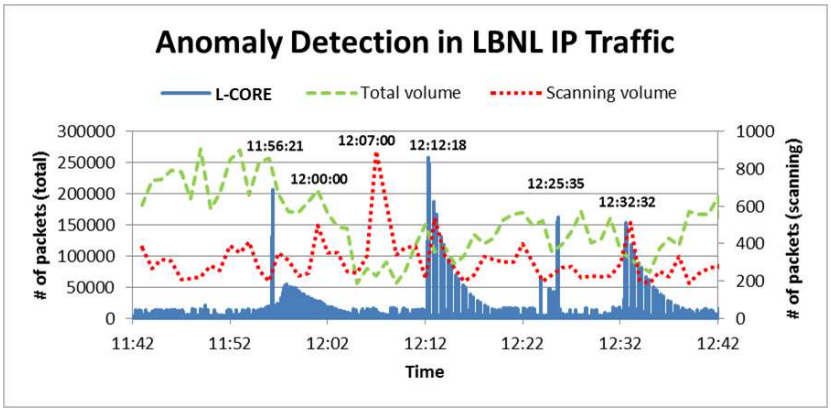


Figure 3.12: Output from the L-CORE algorithm, total network volume, and labeled scanning activity over time. Note the different scales for total volume and scanning activity.

The LBNL dataset was collected by monitoring network flows for IP traffic on a large enterprise network for a period of one hour starting at 11:42am on December 15, 2004 [77]. LBNL researchers then labeled as "scanning activity" any time a single source contacted more than 50 distinct IP addresses in ascending or descending order, as well as activity from two

known internal scanners.

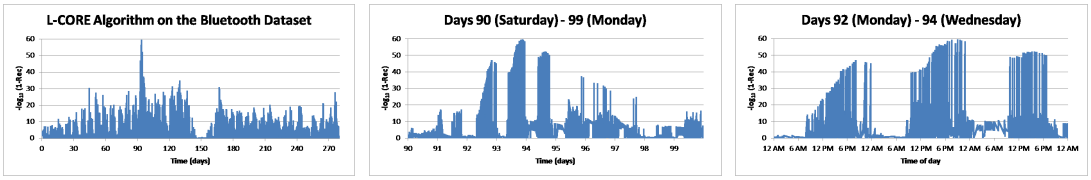
Figure 3.12 shows the output from the L-CORE algorithm – the recency of all IP addresses over time, as well as the total number of packets sent in each minute and the number of those packets that were labeled as scanning activity. We note that spikes in scanning activity do not necessarily coincide with spikes in total volume, and that scanning activity accounts for less than 2% of all traffic even at its peak (thus we use two different axis scales for visualization purposes).

The times with the highest peaks in the L-CORE output are 11:56:21 AM, 12:12:18 PM, 12:25:35 PM and 12:32:32 PM. Three of those immediately precede peaks in labeled scanning activity at 11:57, 12:13, and 12:33 (reflecting the difference in detection latency), but 12:26 does not stand out in terms of total network volume or scanning activity. A closer look at the data reveals that in the three seconds between 12:25:33-12:25:35 PM, a machine with IP address 128.3.204.42 sent packets to over a hundred distinct IP addresses, resulting in 160 of its 214 incident edges having recency values above 0.984. This was not labeled as scanning activity by the LBNL researchers because it did not satisfy their criterion of going through IP addresses monotonically.

On the other hand, the biggest spike in labeled scanning activity, at 12:07 PM, was not reflected in the L-CORE output. Looking at the network trace, the majority of labeled scanning activity overall (over 70% of it) comes from two IP addresses, serving DNS and NBNS requests, respectively. Comparing the period of greatest labeled scanning activity (898 packets from 12:06-12:07 PM) with that of the least (189 packets from 12:38-12:39 PM), more than 500 of the roughly 700 additional packets can be attributed to increased activity at the DNS and NBNS servers during that minute, but still with no more than 10 requests served in any given second. While that does cause an increase in recency at the corresponding nodes, the effect is weaker than for other scenarios with more sudden changes in behavior.

3.3.6 Physical Proximity

The BLUETOOTH dataset consists of communication logs collected from about 100 Bluetooth-enabled mobile devices carried by MIT students and faculty between September 2004 and June 2005 [28]. Each mobile device conducted periodic scans for other nearby devices (with a range



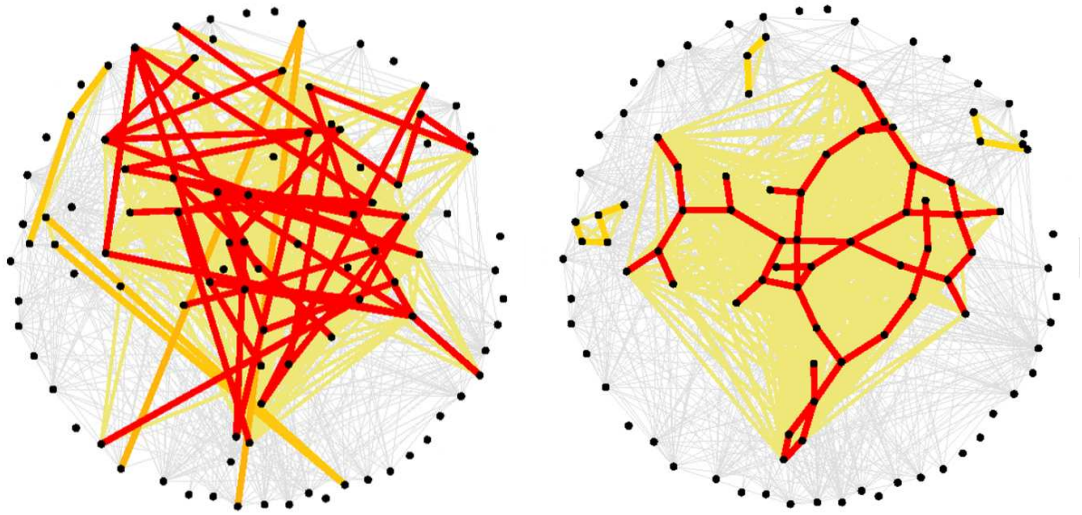
(a) (b) (c)

Figure 3.13: Output from running the L-CORE algorithm on the BLUETOOTH dataset, the recency of all vertices in the network.

of about 5 meters), and recorded the times and device IDs. We cleaned the data to consider only the times at which another device initially came within Bluetooth range, and to ignore consecutive occurrences until the device next became out of range, representing distinct "encounters."

Figure 3.13(a) shows the output from the L-CORE algorithm, the recency of all mobile devices over the duration of the study. Figure 3.13(b) zooms in on a 10-day window with the largest spike. We can see that the detected correlated activity has the highest intensity over a three-day period from Monday to Wednesday, and then lessens drastically, so we further zoom into that time zone in Figure 3.13(c). Matching these dates to MIT's academic calendar from the 2004-2005 academic year, we find that they correspond exactly to the last three days of Fall semester classes.

Since the above output from the L-CORE algorithm only looks at the neighborhoods of individual nodes, we turn to the G-CORE algorithm to give us a better understanding of the kind of interactions that are responsible for the correlated behavior. Figure 3.14(a) visualizes the output from the G-CORE algorithm at 6pm on day 93, around the peak of correlated activity as indicated by the L-CORE results above. We see a giant component containing nearly half of the nodes, with many recent pairwise communications. In contrast, we look at the G-CORE output on a "normal" day, at 12pm on day 100. While the component with highest correlation contains roughly the same number of nodes, there seems to be a significantly lower density of recent communication than on day 93. In fact, examining the recency values for the components of interest indicates that the degree of recent activity exhibited on day 93 is more than one million times less likely to occur by chance than that seen on day 100.



(a) (b)

Figure 3.14: Visualization of output from running the G-CORE algorithm on the Bluetooth dataset at (a) 6pm on day 93, and (b) 12pm on day 100.

3.4 Discussion

3.4.1 Extensions and Applications

The L-CORE algorithm permits streaming analysis, but is limited to detecting correlated activity from a single node. The G-CORE algorithm considers more general subgraph structures, but may not be efficient enough for streaming analysis of large real-world networks. It may be possible to achieve the best of both worlds, however, by channeling the output from the L-CORE algorithm to global algorithms based on the idea of distributed triggers [46]. A central server would choose the sensitivity threshold θ for the L-CORE algorithm and disseminate it to all nodes in the network. Each node would independently monitor its own activity, alerting the central server only when the recency of its outgoing activity exceeds the threshold. More sophisticated methods could then be used to analyze the flagged activity, for example, performing graph algorithms on the subgraph consisting of only those nodes and edges that have been flagged in the past, or dynamically maintaining a sparse data structure such as a minimum spanning tree [43]. In this way, the REWARDS approach can be used as a sampling mechanism to permit more computationally intensive network analysis methods that otherwise would be infeasible on large networks.

With minor modification, the L-CORE and G-CORE algorithms can also be applied to address other network and cyber security challenges. Instead of looking at outgoing edges, a node could compute the recency for *incoming* communication. This could be helpful in early detection of synchronized activity controlled by a botnet, such as a distributed denial-of-service attack. In another setting, neighboring nodes can cooperate to detect anomalous behavior at a node that has been compromised. Alternatively, our algorithms could be used for fault detection by modifying the recency formula to test for a *lack* of recent activity.

In some real-world contexts, message content and other meta-data may be readily available. In such cases, our approach based on temporal dynamics could be complemented with existing techniques that leverage textual, geospatial, or other node or message attributes.

Finally, the REWARDS approach could be generalized to model multiple-recipient emails, public broadcast messages, or bipartite network structures such as those used in many recommendation systems, and to accommodate diurnal patterns or other global trends.

3.4.2 Limitations of Our Approach

Since the REWARDS approach measures recency relative to the previous inter-arrival times of communication activity, our statistical test for correlation will only be effective if the time lag between correlated activity is small compared to the inter-arrival times. In the context of computer network security, for example, with enough foresight a perfect attacker could, from the beginning of its existence, establish a pattern of behavior through which malicious activity could easily be concealed. For example, if a bot master were to send fake messages to all infected hosts in a botnet every minute, it would be easy in the future to broadcast commands undetected. A similar problem is caused by denial-of-service attacks from IP addresses that have never appeared before and thus have no previously established behavior to which to compare.

Another limitation is due to approximating the inter-arrival time distribution. For example, maximum-likelihood parameter estimation for the Bounded Pareto distribution is sensitive to changes in the minimum inter-arrival time. The results from the L-CORE and G-CORE algorithms may also be misleading in cases where the real inter-arrival time distributions do not follow a power law. These limitations may be mitigated by using a more flexible or robust

distribution model. Moreover, we note that these are practical issues specific to the implementation of the inter-arrival time distributions, rather than a theoretical limitation of the REWARDS approach itself.

A broader limitation is that the REWARDS model does not capture temporal phenomena such as *burstiness* and *memory*, which are present in some real-world systems [34]. This is inherent to the approach of modeling activity as being generated by renewal processes, for which inter-arrival times are identically and independently distributed. To address this, more general stochastic models would need to be employed.

3.4.3 Significance and Impact

In this work, we first presented the REWARDS approach, which models network communication between each pair of nodes as a renewal process, and defined a notion of recency that is invariant to changes in time scale, addressing the time-scale bias caused by segmenting time into discrete blocks. To our knowledge, it is the first such solution proposed in the literature.

Next, we proposed statistical methods to measure correlation in recent activity among a subset of nodes in a network, and developed two efficient algorithms to detect correlated events: L-CORE, a streaming algorithm which monitors outgoing activity from a single node; and G-CORE, a global algorithm which can detect correlated activity in disparate parts of the network simultaneously.

A strength of the REWARDS approach is that it only requires information about the times of communication, and does not rely on message content. This makes it particularly useful for applications in which message content may be unavailable or encrypted, or when data privacy is a concern.

Our approach is also well-suited for interfacing with a human analyst. Both the L-CORE and G-CORE algorithms explicitly output the set of events that constitutes each instance of correlated activity, along with the recency value which indicates the strength of the correlation. This can guide a human analyst in selecting and prioritizing nodes and events for deeper scrutiny.

Correlated event detection is just one application of the REWARDS model. Our novel approach to representing and analyzing activity in event-driven networks could also be leveraged

to capture long-term patterns of correlation and dependence between processes, leading to potential uses for studying information diffusion and influence in networks. We explore these and other directions for future work in Chapter 6.

Chapter 4

Discovering Functional Communities

4.1 Introduction

4.1.1 Background and Motivation

Many networks can be seen as an interface between individuals and content, which may be material or conceptual. In social media, content is shared by individuals to an audience that has subscribed to received it, or is otherwise made publicly accessible. Some prominent examples are web logs (blogs), internet forums, and the microblogging service Twitter. In other contexts, individuals publicly endorse companies, products, artists, or audio and video content that they like. Examples include Amazon.com, YouTube, and Facebook Pages.

Such relationships can be represented as a matrix with rows corresponding to individuals in the network and columns corresponding to content, as in Figure 4.1. A dense rectangular block in the matrix indicates a *functional community*, a group of individuals who share or endorse common content. For example, if the matrix represents a restaurant recommendation network, a functional community could consist of lovers of Italian cuisine along with their favorite Italian restaurants.

However, since the order of the rows and columns of a matrix are arbitrary, functional communities may not be readily apparent as contiguous blocks. Rather, any *bicluster* – a pair of row and column subsets – could form a functional community. Furthermore, individuals may have multiple interests and therefore belong to more than one community. Efficiency is also a concern, since real-world networks may contain many individuals and many distinct pieces of content.

A task known as *co-clustering* – simultaneously permuting and clustering the rows and columns of the matrix – can facilitate the efficient discovery of dense biclusters, and allows for

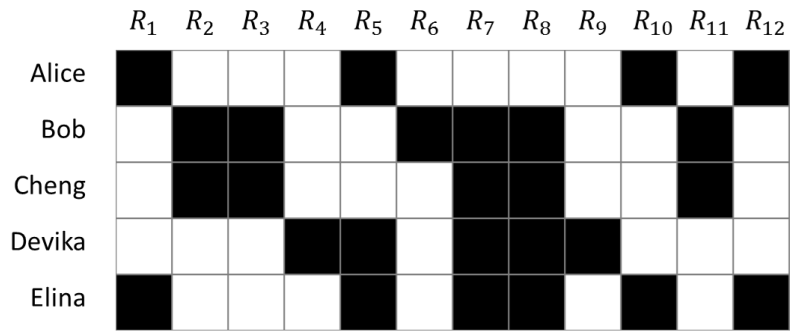


Figure 4.1: A matrix representing relationships between individuals and content in a network.

multiple community membership. We seek an algorithm that does not rely on domain-specific knowledge or data-specific parameters so it can be easily applied to a variety of networks. Before presenting our own approach, we provide a survey of related work.

4.1.2 Related Work

There is a significant amount of related work in the data mining, machine learning, databases, and bioinformatics literature – each motivated by different applications and with different goals in mind. These tasks may be variously referred to in the literature as co-clustering, biclustering, two-mode clustering, or matrix block partitioning.

In bioinformatics, biclustering algorithms have been developed to find patterns in gene expression [71, 93, 30]. A matrix is constructed where each row is a gene, each column is an experiment under different conditions, and each entry indicates the level of expression of that gene under those conditions. Matrices are dense and real-valued, and the goal is to identify genes that behave similarly, demonstrated by expression levels that have a constant, linear, or multiplicative relationship across similar experimental conditions. Due to the density of the matrices and the more intricate bicluster structure desired, algorithms in this domain tend to find only one bicluster at a time, and efficiency concerns frequently restrict their use to smaller datasets.

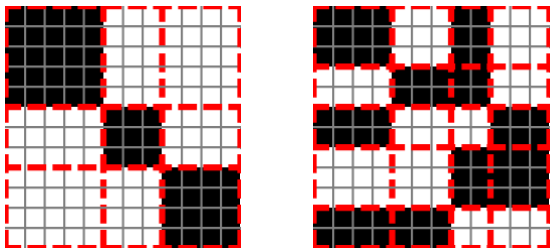
Algorithms for finding structure in matrices can be used to improve the efficiency of database storage and querying. Navathe et al. address the problem of vertical partitioning of a database, which groups attribute columns that frequently need to be accessed together [73]. Muthukrishnan et al. consider the problem of rectangular partitioning of a matrix, partitioning the matrix

into a set of rectangular tiles without permuting the rows or columns, which is useful for maintaining multi-dimensional histograms [72]. They consider a variety of evaluative metrics, and give heuristics and complexity theoretic results for several related optimization problems. For the applications we are interested in, row and column permutations are allowed, and are frequently necessary to discover latent structure.

Co-clustering has also been found to improve upon standard techniques for clustering unipartite data. Such approaches entail first computing a similarity matrix, a symmetric square matrix indicating the similarity of each pair of points in the dataset, to which co-clustering algorithms are then applied. A similar approach can be employed for *graph partitioning*, where the goal is to cluster the vertices in the graph so that there is a high density of edges within clusters and few edges going between them, a task which has applications to community discovery and parallel computing for graph algorithms. Here, co-clustering algorithms would be applied to the adjacency matrix.

In these scenarios, co-clustering is performed on a square matrix, where the rows and the columns represent the same set of objects. The desired result is a partitioning of the objects, which corresponds to a *block diagonal* structure in the matrix, like that seen in Figure 4.2(a). Numerous algorithms to achieve this have been proposed in the literature based on linear algebraic techniques such as singular value decomposition, referred to broadly as *spectral clustering* [70]. Further works have suggested algorithms which permit a slightly more general matrix structure, such as block diagonal with overlap [50], block tridiagonal [88], or rectangular matrices [56]. In some domains, however, matrices arise with more varied block structures, such as that in Figure 4.2(b), which would not be captured well by these methods.

Another approach is to first define a metric over co-clusterings, and then search for a co-clustering which optimizes the metric. Dhillon et al. define a metric based on mutual information [25]. Their algorithm optimizes the metric subject to a constraint on the number of row and column clusters. On the other hand, our approach is parameter-free, searching over partitions with varying numbers of clusters and implicitly determining the number of clusters that yields the best result.



(a) (b)

Figure 4.2: Two matrices with clear block structures. Spectral clustering methods are suited well for block diagonal matrices like that in (a), but less so for matrices like that in (b). The dashed lines suggest a good co-clustering.

Chakrabarti et al. follow a similar approach, but are motivated by the task of matrix compression, and propose *encoding cost* as a metric, the number of bits required to represent a matrix [16]. They present the Cross-Association algorithm, a heuristic for finding a co-clustering with minimal encoding cost, incrementally increasing the number of row or column clusters until a local optimum is reached. It is parameter-free, a strength over approaches which require the number of clusters to be known in advance. However, the efficiency of their algorithm depends on quick convergence to a small number of clusters. This may be effective for achieving their objective of a good compression ratio, but may run contrary to our goal of finding dense biclusters, especially in the case of very large and sparse matrices. Furthermore, we found that in practice, since the algorithm alternates between refining the row and column clusters instead of doing both simultaneously, it may get stuck at a local optimum because neither the rows nor the columns alone can sufficiently distinguish clusters. A simple example of this is a matrix where all rows have the same density, and likewise for the columns – the algorithm will never progress past a single row and column cluster.

4.1.3 Contributions and Outline

Our contributions can be summarized as follows:

- Two intuitive properties of co-clustering metrics that aim to reward large, dense biclusters
- A class of metrics which uniquely satisfy those properties among known metrics
- The CC-MACS algorithm, an efficient heuristic algorithm to find a good co-clustering of an $m \times n$ matrix in $O\left(N \cdot \max\left(\log(mn), \log^2\left(\frac{mn}{N}\right)\right)\right)$ time, where N is the number of

non-zeros in the matrix

In this work, we break away from the traditional mindset that a good co-clustering must consist of a small number of clusters. While it may be true for applications such as matrix compression that "fewer is better," in the context of community discovery that is not necessarily the case. Instead of having a small number of very large blocks, CC-MACS may return a co-clustering with hundreds or thousands of blocks, among which will be the dense biclusters corresponding to functional communities. To our knowledge, this idea is novel to our approach, and allows us to out-perform techniques based on low-dimensional approximations or requiring the maximum number of clusters to be specified in advance. The CC-MACS algorithm is also designed to leverage the sparsity of many real-world datasets, running in time *sub-linear in the size of the matrix* for sparse matrices.

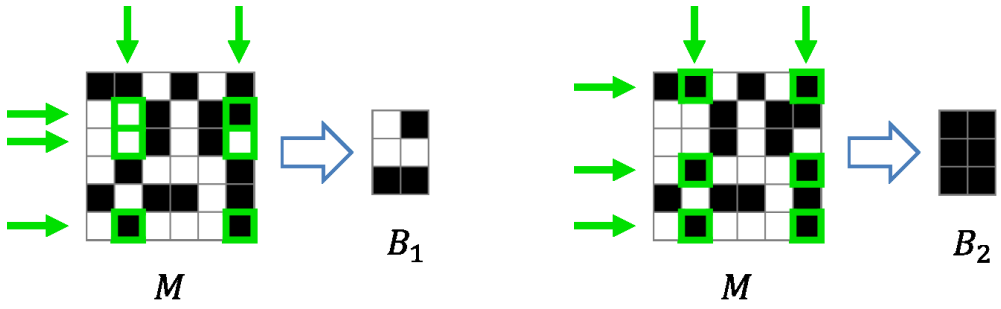
In particular, our methods have the following benefits over previously proposed approaches: (1) the dense biclusters in the matrix need not have a block diagonal structure; (2) our algorithm explores the breadth of the search space rather than getting stuck at local optima; (3) the results are not dependent on user-specified parameters; and (4) our algorithm is sub-linear in the size of the matrix for $N \ll mn$, making it extremely efficient for large, sparse datasets.

In Section 4.2.1, we give preliminary definitions and the framework for our approach. Section 4.2.2 addresses the question of choosing an appropriate metric. We present the CC-MACS (Co-Clustering via Maximal Anti-Chain Search) algorithm in Section 4.2.3 and analyze its running time in Section 4.2.4. In Section 4.3, we evaluate our approach with experiments on synthetic and real-world datasets. In Section 4.4, we conclude with discussion of the strengths and limitations of our approach, the significance of our work, and directions for future work.

4.2 Methodology

4.2.1 Preliminaries

Let M be an $m \times n$ matrix. A *bicluster* of M is a subset of matrix entries formed by the intersection of a set of rows $I \subseteq [m]$ and a set of columns $J \subseteq [n]$, and is denoted by $M_{I,J}$. We define the weight of a bicluster $B = M_{I,J}$ to be $w(B) = \sum_{i \in I,\ j \in J} M_{i,j}$; the area $a(B) = |I| \cdot |J|$; the semiperimeter $s(B) = |I| + |J|$; and the density $d(B) = w(B)/a(B)$.



(a) (b)

Figure 4.3: Two biclusters in a matrix.

Figure 4.3 shows two different biclusters in the same matrix. Bicluster B_2 is arguably better than B_1 since it is the same size but more dense, indicating a stronger association between the corresponding rows and columns.

Let $\beta(M)$ denote the set of all possible biclusters of M. A *bicluster partition* of M is a set of biclusters $\Pi \subseteq \beta(M)$ such that each element $M_{i,j}$ is contained in exactly one bicluster. *Co-clustering* is the data mining task of simultaneously clustering the rows and columns of M, which naturally corresponds to a bicluster partition of M.

Our approach consists of two main components: (1) define a quality metric for bicluster partitions; and (2) find a co-clustering that maximizes the value of the metric. We address each of these tasks in the following sections.

4.2.2 Choosing a Metric

Here we consider metrics to evaluate the quality of a bicluster partition. A *bicluster partition metric* μ is a mapping from bicluster partitions to real values, i.e. $\mu(\Pi) \in \mathbb{R}$ where Π is a bicluster partition.

A variety of bicluster partition metrics have been proposed in the literature. To decide which are most appropriate for our context, we first suggest two desirable properties, motivated by our goal of identifying large, dense biclusters. Figure 4.4 gives motivating examples of properties **P1** and **P2**.

(P1) Merging a positive-weight bicluster with a zero-weight bicluster decreases the value of the metric.

(a) (b)

Figure 4.4: Examples illustrating properties (a) **P1** and (b) **P2**.

(P2) Merging two non-empty biclusters of the same density increases the value of the metric.

We now propose a class of metrics that satisfy both properties:

$$\left\{ \mu_\alpha(\Pi) = \sum_{B \in \Pi} \frac{a(B)^2}{s(B)} \cdot d(B)^{2+\alpha} : \alpha \geq 0 \right\}$$

The intuition is that the first term favors larger biclusters and the second term favors denser biclusters, so overall the metric favors partitions containing biclusters that are both large and dense. The value of the parameter α can be used to balance the trade-off between size and density of the biclusters.

Theorem 4.1. *For all $\alpha \geq 0$, μ_α satisfies property* **P1**.

Proof. Consider two biclusters B_1 and B_2. For $B_1 \cup B_2$ to also be a bicluster, it must be that B_1 and B_2 share either the same set of rows or the same set of columns. Without loss of generality, suppose they share the same set of r rows; then their column sets must be disjoint. Let $w(B_1) = w_1$ and $w(B_2) = 0$, and let B_1 and B_2 have c_1 and c_2 columns, respectively. Table 4.1 gives several values used in the computation of μ_α.

Bicluster	Weight	Area	Semiperimeter
B_1	w_1	$r \cdot c_1$	$r + c_1$
B_2	0	$r \cdot c_2$	$r + c_2$
$B_1 \cup B_2$	w_1	$r \cdot (c_1 + c_2)$	$r + c_1 + c_2$

Table 4.1: Values used in the proof that μ_α satisfies P1.

We want to show:

$$\begin{aligned}
\mu_\alpha(\{B_1, B_2\}) &> \mu_\alpha(\{B_1 \cup B_2\}) \\
\frac{(rc_1)^2}{r+c_1} \cdot \left(\frac{w_1}{rc_1}\right)^{2+\alpha} + \frac{(rc_2)^2}{r+c_2} \cdot \left(\frac{0}{rc_1}\right)^{2+\alpha} &> \frac{(r(c_1+c_2))^2}{r+c_1+c_2} \cdot \left(\frac{w_1}{r(c_1+c_2)}\right)^{2+\alpha} \\
\frac{(rc_1)^2}{r+c_1} \cdot \left(\frac{w_1}{rc_1}\right)^{2+\alpha} &> \frac{(r(c_1+c_2))^2}{r+c_1+c_2} \cdot \left(\frac{w_1}{r(c_1+c_2)}\right)^{2+\alpha} \\
\frac{{w_1}^{2+\alpha}}{(r+c_1)(rc_1)^\alpha} &> \frac{{w_1}^{2+\alpha}}{(r+c_1+c_2)(r(c_1+c_2))^\alpha} \\
(r+c_1+c_2)(r(c_1+c_2))^\alpha &> (r+c_1)(rc_1)^\alpha
\end{aligned}$$

Each consecutive statement is true if and only if the preceeding statement is true by simple algebraic manipulation, and the last statement is true since $c_2 > 0$ and $\alpha \geq 0$. Therefore, μ_α satisfies property **P1**. □

Theorem 4.2. *For all $\alpha \geq 0$, μ_α satisfies property* **P2**.

Proof. Consider two biclusters B_3 and B_4. For $B_3 \cup B_4$ to also be a bicluster, it must be that B_3 and B_4 share either the same set of rows or the same set of columns. Without loss of generality, suppose they share the same set of r rows; then their column sets must be disjoint. Let $d(B_3) = d(B_4) = d$, and let B_3 and B_4 have c_3 and c_4 columns, respectively. Table 4.2 gives several values used in the computation of μ_α.

Bicluster	Density	Area	Semiperimeter
B_3	d	$r \cdot c_3$	$r + c_3$
B_4	d	$r \cdot c_4$	$r + c_4$
$B_3 \cup B_4$	d	$r \cdot (c_3 + c_4)$	$r + c_3 + c_4$

Table 4.2: Values used in the proof that μ_α satisfies P2.

We want to show:

$$\begin{aligned}
\mu_\alpha(\{B_3, B_4\}) &< \mu_\alpha(\{B_3 \cup B_4\}) \\
\frac{(rc_3)^2}{r+c_3} \cdot d^{2+\alpha} + \frac{(rc_4)^2}{r+c_4} \cdot d^{2+\alpha} &< \frac{(r(c_3+c_4))^2}{r+c_3+c_4} \cdot d^{2+\alpha} \\
\frac{{c_3}^2}{r+c_3} + \frac{{c_4}^2}{r+c_4} &< \frac{(c_3+c_4)^2}{r+c_3+c_4} \\
\frac{{c_3}^2(r+c_4) + {c_4}^2(r+c_3)}{(r+c_3)(r+c_4)} &< \frac{(c_3+c_4)^2}{r+(c_3+c_4)} \\
\frac{r({c_3}^2+{c_4}^2) + c_3c_4(c_3+c_4)}{r^2 + r(c_3+c_4) + c_3c_4} &< \frac{(c_3+c_4)^2}{r+(c_3+c_4)}
\end{aligned}$$

$$\begin{aligned}\left(r({c_3}^2+{c_4}^2)+c_3c_4(c_3+c_4)\right)\cdot(r+(c_3+c_4)) \quad &< \\ &\left((c_3+c_4)^2\right)\cdot\left(r^2+r(c_3+c_4)+c_3c_4\right)\end{aligned}$$

$$\begin{aligned}r^2({c_3}^2+{c_4}^2)+r(c_3+c_4)({c_3}^2+c_3c_4+{c_4}^2)+c_3c_4(c_3+c_4)^2 \quad &< \\ &r^2(c_3+c_4)^2+r(c_3+c_4)^3+c_3c_4(c_3+c_4)^2\end{aligned}$$

$$\begin{aligned}r^2({c_3}^2+{c_4}^2)+r(c_3+c_4)({c_3}^2+c_3c_4+{c_4}^2)+c_3c_4(c_3+c_4)^2 \quad &< \\ &r^2({c_3}^2+2c_3c_4+{c_4}^2)+r(c_3+c_4)({c_3}^2+2c_3c_4+{c_4}^2)+c_3c_4(c_3+c_4)^2\end{aligned}$$

$$0 \quad < \quad r^2(2c_3c_4)+r(c_3+c_4)(c_3c_4)$$

Each consecutive statement is true if and only if the preceeding statement is true by simple algebraic manipulation, and the last statement is true since $r, c_1, c_2 > 0$. Therefore, μ_α satisfies property **P2**. □

We now examine which of these properties are satisfied by several previously proposed and baseline metrics in Table 4.3. That none of the previously proposed metrics examined here satisfy both properties does not imply that they are bad metrics. Rather, they were proposed with different objectives in mind. Properties **P1** and **P2** were motivated by our goal of finding large, dense biclusters. Other objectives could be optimal compression of the matrix (inverse encoding cost) or non-uniformity of densities of the biclusters (Kullback-Liebler divergence from uniform).

Metric	**P1**	**P2**
Inverse encoding cost [16]	–	✓
Kullback-Leibler divergence from uniform	✓	–
$\sum_{B\in\Pi} w(B)$	–	–
$\sum_{B\in\Pi} d(B)$	✓	–
$\sum_{B\in\Pi} w(B)\cdot d(B)$	–	✓
$\sum_{B\in\Pi}(a(B)^2/s(B))\cdot d(B)^{2+\alpha}$	✓	✓

Table 4.3: Properties satisfied by several metrics.

We note that our proposed metrics take the form $\sum_{B\in\Pi} f(B)$, where $f : \beta(M) \to \mathbb{R}$ is a function of the weight and dimensions of a bicluster. The $\sum$ operator is both commutative

and associative, so our metrics can be computed by combining the f-values for the biclusters in any order. In the next section, we leverage this property to develop more efficient algorithms to find good bicluster partitions.

4.2.3 The CC-MACS Algorithm

We now present the CC-MACS (Co-Clustering via Maximal Anti-Chain Search) algorithm, which efficiently searches for a good co-clustering according to a given metric. We first note that the total number of possible co-clusterings of an $m \times n$ matrix is exponential in the size of the matrix (the product of the mth and nth Bell numbers), so an exhaustive search is infeasible. Our strategy for overcoming this computational challenge is to first build trees on the rows and columns, respectively, and then to consider only co-clusterings corresponding to maximal anti-chains in the trees.

A *maximal anti-chain* of a rooted tree is a maximal set of nodes in the tree, none of which is a descendant of any other. For example, the blue nodes in each of the trees in Figures 4.5(c)-(i) form a maximal anti-chain. Note that the subtrees of the nodes in a maximal anti-chain correspond to a partition of the leaves of the tree. Therefore any pair of maximal anti-chains of the row and column trees, respectively, corresponds to a co-clustering of the matrix.

This is still a computational challenge, however, because there are $\Omega(2^n)$ maximal anti-chains in a complete binary tree with n leaves. We employ a heuristic to find the most likely candidates by traversing the row and column trees simultaneously, starting at the leaves and greedily merging the nodes that result in the greatest increase in the metric value. Figure 4.5 illustrates an example run of the CC-MACS algorithm. Pseudocode is given in Algorithm 4.1.

4.2.4 Complexity Analysis

Consider an $m \times n$ matrix containing N non-zero entries. We first make the following claim about the running time of Step 4 of the CC-MACS algorithm.

Theorem 4.3. *The arrays W and F populated in Step 4 of the CC-MACS algorithm contain $O\left(N \cdot \log^2\left(\frac{mn}{N}\right)\right)$ non-zero entries.*

Proof. Let M denote the original matrix, and let T^{row} and T^{col} denote the k-d trees constructed

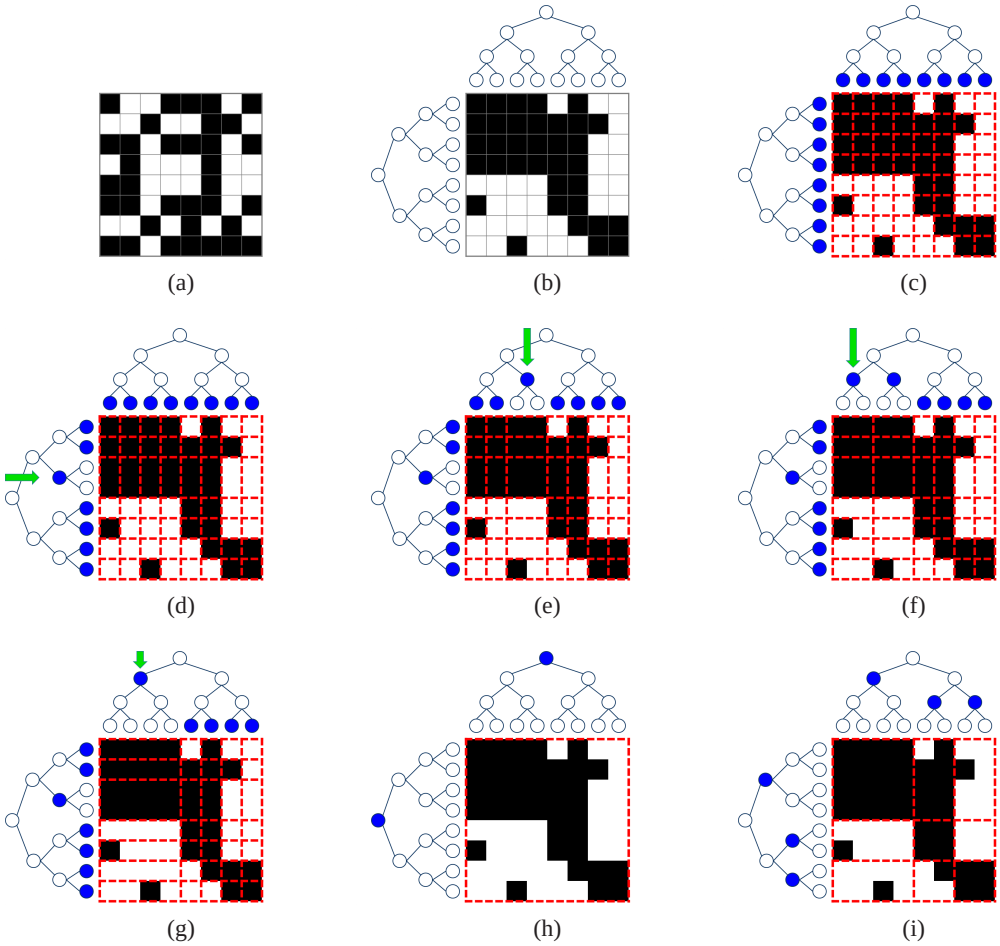


Figure 4.5: Illustration of the CC-MACS algorithm. Figure (a) shows an example matrix as input to the algorithm. Figure (b) shows the same matrix permuted to reflect the k-d trees constructed in Steps 2 and 3. The maximal anti-chains S^{row} and S^{col} are indicated by the blue nodes, initially set to be the leaves in Step 5 (Figure (c)), and dynamically updated during the loop in Step 8 (Figures (d)-(g)). Figure (h) shows the co-clustering after the final iteration of the loop, when S^{row} and S^{col} are the roots of the trees, corresponding to the single bicluster consisting of the entire matrix. Figure (i) shows the result of the CC-MACS algorithm, the co-clustering that was found to maximize the metric value, indicated by the red lines.

Algorithm 4.1 The CC-MACS Algorithm (Co-Clustering via Maximal Anti-Chain Search)

Input: An $m \times n$ matrix M and a bicluster partition metric $\mu(\Pi) = \bigoplus_{B \in \Pi} f(B)$, where $\bigoplus$ is a commutative and associative binary operator, and $f : \beta(M) \to \mathbb{R}$ is a function of the weight and dimensions of a bicluster.

Output: A co-clustering Π of M.

1: Initialize partitions Π^{row} and Π^{col} to be the sets of singletons of the rows and columns of M, respectively.

2: Construct a k-d tree T^{row} over the vector sums corresponding to row clusters in Π^{row}, after first applying a random projection, as in [49]. Let $L(T^{row})$ denote the set of leaves of T^{row}, and let $I_x \subseteq [m]$ denote the indices of rows in the subtree rooted at node $x \in T^{row}$.

3: Construct a k-d tree T^{col} over the vector sums corresponding to column clusters in Π^{col}, similarly to above. Let $L(T^{col})$ denote the set of leaves of T^{col}, and let $J_y \subseteq [n]$ denote the indices of columns in the subtree rooted at node $y \in T^{col}$.

4: Populate a two-dimensional array W indexed by nodes in T^{row} and T^{col}, respectively, where entry $W[x, y]$ is the number of non-zeros in the bicluster M_{I_x, J_y}. From this, populate another array F of the same dimensions, containing values $F[x, y] = f(M_{I_x, J_y})$.

5: Let S^{row}, S^{row}_{max} and S^{col}, S^{col}_{max} be maximal anti-chains over T^{row} and T^{col}, respectively. Initialize them as $S^{row} = S^{row}_{max} = L(T^{row})$, $S^{col} = S^{col}_{max} = L(T^{col})$.

6: Maintain the current and maximum metric values, μ_{curr} and μ_{max}, and initialize them for the partition corresponding to the current $S^{row} \times S^{col}$.

7: Maintain a max heap H^{row} containing only nodes $x \in T^{row}$ such that both x.LEFT and x.RIGHT are in S^{row}, with priorities $h^{row}(x) = \sum_{y \in S^{col}} F[x, y] - f[x.\text{LEFT}, y] - f[x.\text{RIGHT}, y]$, the marginal value from including x in a maximal anti-chain instead of its children. Construct H^{col} similarly.

8: While at least one of H^{row} and H^{col} is non-empty (without loss of generality, suppose that $H^{row}.maxPriority() \geq H^{col}.maxPriority()$):
Update the dynamic data structures and variables:

- $x \leftarrow H^{row}.deletemax()$;
- $S^{row} \leftarrow S^{row} + x - x.\text{LEFT} - x.\text{RIGHT}$;
- $\mu_{curr} \leftarrow \mu_{curr} + h^{row}(x)$;
- Update $h^{col}(y)$ for each $y \in H^{col}$;

If $x.\text{SIBLING} \in S^{row}$, then $H^{row}.add(x.\text{PARENT})$;
If $\mu_{curr} \geq \mu_{max}$, perform the following updates:

- $\mu_{max} \leftarrow \mu_{curr}$;
- $S^{row}_{max} \leftarrow S^{row}$;

9: Update Π^{row} and Π^{col} to be the row and column partitions corresponding to nodes in S^{row}_{max} and S^{col}_{max}, respectively.

10: Repeat Steps 2-9 using the updated Π^{row} and Π^{col}. Continue while at least one is updated.

11: Return $\Pi = \Pi^{row} \times \Pi^{col}$, the co-clustering formed by the intersection of Π^{row} and Π^{col}.

on the rows and columns, respectively. For a pair of nodes $x \in T^{row}$ and $y \in T^{col}$, the array entry $W[x, y]$ (and therefore also $F[x, y]$) is non-zero only if the bicluster M_{I_x, J_y} has at least one non-zero entry.

Let T_l denote the set of nodes in level l of tree T. The critical observation to make is that for all $y \in T^{col}$,

$$\begin{aligned} |\{x \in T_l^{row} : W[x, y] > 0\}| &\leq \left|\left\{x \in T_{l+1}^{row} : W[x, y] > 0\right\}\right| \\ &\leq |\{x \in L\left(T^{row}\right) : W[x, y] > 0\}|. \end{aligned} \tag{4.1}$$

The first inequality follows from the fact that

$$W[x, y] > 0 \implies W[x.left, y] > 0 \text{ or } W[x.right, y] > 0,$$

and the latter by induction. We also have the trivial bound

$$|\{x \in T_l^{row} : W[x, y] > 0\}| \leq |\{x \in T_l^{row}\}| = 2^l. \tag{4.2}$$

When less than half of the nodes $x \in T_{l+1}^{row}$ have $W[x, y] > 0$, then bound 4.1 is better; otherwise, bound 4.2 is better. The following analysis finds the optimal level at which to switch, yielding a bound on the overall number of non-zero entries.

Suppose we use bound 4.2 for the top l^* levels of T^{row}, and bound 4.1 for the remaining $\log(m) - l^*$ levels. Then we have the following bound on the number of non-zero entries across

all nodes in T^{row} and all leaves in T^{col}:

$$\begin{aligned}
&\left|\left\{(x,y) \in T^{row} \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad = \sum_{l=1}^{l^*} \left|\left\{(x,y) \in T_l^{row} \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\qquad\quad + \sum_{l=l^*+1}^{\log(m)} \left|\left\{(x,y) \in T_l^{row} \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad = \sum_{l=1}^{l^*} \sum_{y \in L\left(T^{col}\right)} \left|\{x \in T_l^{row} : W[x,y] > 0\}\right| \\
&\qquad\quad + \sum_{l=l^*+1}^{\log(m)} \sum_{y \in L\left(T^{col}\right)} \left|\{x \in T_l^{row} : W[x,y] > 0\}\right| \\
&\quad \leq \sum_{l=1}^{l^*} \sum_{y \in L\left(T^{col}\right)} 2^l + \sum_{l=l^*+1}^{\log(m)} \sum_{y \in L\left(T^{col}\right)} \left|\{x \in L\left(T^{row}\right) : W[x,y] > 0\}\right| \\
&\qquad\quad \text{(by bounds 4.1 and 4.2 above)} \\
&\quad = \sum_{y \in L\left(T^{col}\right)} \sum_{l=1}^{l^*} 2^l + \sum_{l=l^*+1}^{\log(m)} \left|\left\{(x,y) \in L\left(T^{row}\right) \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad = n \cdot \left(2^{l^*+1} - 1\right) + \left(\log(m) - l^*\right) \cdot N
\end{aligned}$$

To get the best bound possible, we optimize over l^*. We set the derivative of the above expression equal to 0 to find the critical values:

$$\begin{aligned}
\frac{d}{d\,l^*}\left(n \cdot \left(2^{l^*+1} - 1\right) + \left(\log(m) - l^*\right) \cdot N\right) &= n \cdot 2^{l^*+1} \cdot \ln(2) - N = 0 \\
\Longrightarrow l^* &= \log\left(\frac{N}{2 \cdot \ln(2) \cdot n}\right)
\end{aligned}$$

This minimizes the function, giving an optimal bound of

$$\begin{aligned}
&\left|\left\{(x,y) \in T^{row} \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad \leq n \cdot \left(2 \cdot \frac{N}{2 \cdot \ln(2) \cdot n} - 1\right) + \log\left(\frac{2 \cdot \ln(2) \cdot mn}{N}\right) \cdot N \\
&\quad = \frac{N}{\ln(2)} - n + N \cdot \log\left(\frac{2 \cdot \ln(2) \cdot mn}{N}\right) \\
&\quad = O\left(N \cdot \log\left(\frac{mn}{N}\right)\right).
\end{aligned}$$

This gives a bound on the number of non-zeros across all nodes in T^{row}, but only the leaves in T^{col}. An analytical approach analogous to that applied for the row nodes above applies for

the column nodes as well, giving a final bound across all nodes in T^{row} and T^{col}:

$$\left|\left\{(x,y) \in T^{row} \times T^{col} : W[x,y] > 0\right\}\right| = O\left(N \cdot \log^2\left(\frac{mn}{N}\right)\right),$$

as desired. □

In fact, we note that the bound in Theorem 4.3 is tight.

Theorem 4.4. *There exists a constant c such that for all $m, n, N \leq mn \in \mathbb{N}$, there exists an $m \times n$ matrix M with N non-zero entries for which the corresponding array W contains at least $c \cdot N \cdot \log^2\left(\frac{mn}{N}\right)$ non-zero entries.*

Proof. For simplicity, assume that $d = \sqrt{\frac{mn}{N}}$ is an integer. Consider the $m \times n$ matrix M constructed as follows:

$$M_{i,j} = \begin{cases} 1 & \text{if } i \equiv 0 \pmod{d} \text{ and } j \equiv 0 \pmod{d} \\ 0 & \text{otherwise} \end{cases}$$

First, we note that M has $\frac{m}{d} \cdot \frac{n}{d} = \sqrt{\frac{mN}{n}} \cdot \sqrt{\frac{nN}{m}} = N$ non-zero entries. Next, consider a node $x \in T^{row}_{l^*}$, where $l^* = \log(m) - \log(d)$. The subtree rooted at x has height $\log(d)$, and therefore contains $2^{\log(d)} = d$ leaves. By construction, exactly one of those leaves corresponds to a row with non-zero entries, so we have a bijection from non-zero leaves to non-zero nodes in each of levels $T^{row}_{\log(m)-\log(d)}, \ldots, T^{row}_{\log(m)-1}$. (If d is not an exact power of 2, the analysis is still accurate to within a factor of 2.) Summing over all leaves $y \in L\left(T^{col}\right)$, corresponding to columns of M, we have that

$$\begin{aligned}
&\left|\left\{(x,y) \in T^{row} \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad \geq \sum_{l=\log(m)-\log(d)}^{\log(m)} \left|\left\{(x,y) \in T^{row}_l \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad = \sum_{l=\log(m)-\log(d)}^{\log(m)} \left|\left\{(x,y) \in L\left(T^{row}\right) \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad = \log(d) \cdot \left|\left\{(x,y) \in L\left(T^{row}\right) \times L\left(T^{col}\right) : W[x,y] > 0\right\}\right| \\
&\quad = \log(d) \cdot N \quad \text{(corresponding to the } N \text{ non-zero entries in } M\text{)}.
\end{aligned}$$

This counts the number of non-zeros in W across all nodes in T^{row}, but only the leaves in T^{col}. Similar analysis to that above gives a final bound across all nodes in T^{row} and T^{col}:

$$\left|\left\{(x,y) \in T^{row} \times T^{col} : W[x,y] > 0\right\}\right| \geq N \cdot \log^2(d) = N \cdot \log^2\left(\frac{mn}{N}\right),$$

as desired. □

We now proceed to analyze the running time of the entire CC-MACS algorithm.

Step 1, initializing the partitions, takes $O(m+n)$ time. The m rows can be projected onto a $\log(m)$-dimensional space in $O(N \log m)$ time, after which the k-d tree on the rows can be computed in $O(m \log m)$ time; likewise, the n rows can be projected onto a $\log(n)$-dimensional space in $O(N \log n)$ time, after which the k-d tree on the columns can be computed in $O(n \log n)$ time; so Steps 2 and 3 take $O(N \cdot (\log m + \log n)) = O(N \log(mn))$ time total. Using dynamic programming, the arrays W and F in Step 4 can be populated in time linear in the number of non-zeros in the resulting arrays, which is $O\left(N \cdot \log^2\left(\frac{mn}{N}\right)\right)$ by Theorem 4.3. Step 5 is $O(m+n)$, and Step 6 is $O(N)$. Computing the priority values in Step 7 takes $O(N)$ time. Inserting and deleting the elements in the row and column heaps in Steps 7-8 takes $O(m \log m + n \log n)$ time total since each node is inserted and deleted at most once. Updating $h^{col}(y)$ in Step 8 takes $O(\log n)$ time for each $y \in H^{col}$, which is performed for each iteration of the loop where an element $x \in H^{row}$ was chosen such that $F[x,y]$ is non-zero, for a total of $O(N \log n)$; the total for iterations where an element from H^{col} was chosen is $O(N \log m)$. In total, the CC-MACS algorithm runs in $O\left(\kappa \cdot N \cdot \max\left(\log(mn), \log^2\left(\frac{mn}{N}\right)\right)\right)$ time, where κ is the number of iterations of Step 10.

4.3 Evaluation

We first evaluate the effectiveness of the CC-MACS algorithm for finding dense biclusters by comparing with existing and baseline co-clustering algorithms. Then we use the CC-MACS algorithm to identify functional communities in social media. These are the algorithms we use in our experiments:

- CC-MACS algorithm with $\mu_2 = \sum_{B \in \Pi} \frac{a(B)^2}{s(B)} \cdot d(B)^4$
- CC-MACS algorithm with $\mu_1 = \sum_{B \in \Pi} \frac{a(B)^2}{s(B)} \cdot d(B)^3$

- CC-MACS algorithm with $\mu_0 = \sum_{B \in \Pi} \frac{a(B)^2}{s(B)} \cdot d(B)^2$
- Cross-Association: minimizes encoding cost [16]
- One-block: consists of a single large bicluster
- Singletons: each matrix entry is in its own bicluster

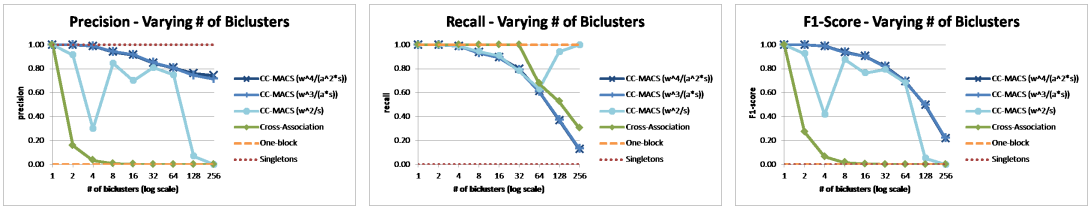
4.3.1 Prediction Accuracy

First we outline an unsupervised learning task by which to evaluate co-clustering algorithms. Given a matrix M containing an unknown set of possibly noisy biclusters $\{B_k\}_{k \in \mathbb{N}}$, co-cluster M such that elements of bicluster B_k only appear in the same block in the partition as other elements of B_k. That is, the pair of matrix entries $(a_1, a_2) = (M_{i_1,j_1}, M_{i_2,j_2})$ form a positive instance if $(\exists k)\ a_1, a_2 \in B_k$; and a negative instance if at least one of a_1, a_2 appears in a bicluster and $(\nexists k)\ a_1, a_2 \in B_k$.

We perform experiments to evaluate the accuracy of several co-clustering algorithms for the learning task described above using synthetically generated matrices with a variety of parameters. Specifically, $\text{MGEN}(m, n, k, r, s, p)$ generates an $m \times n$ matrix M with k biclusters of size $r \times s$ selected randomly from M, where each non-bicluster entry is a 0, and each bicluster entry is a 1 with probability $1 - p$. For each co-clustering algorithm, we compute the precision, recall, and F_1-score (a statistical measure that considers both precision and recall), averaged over 10 trials.

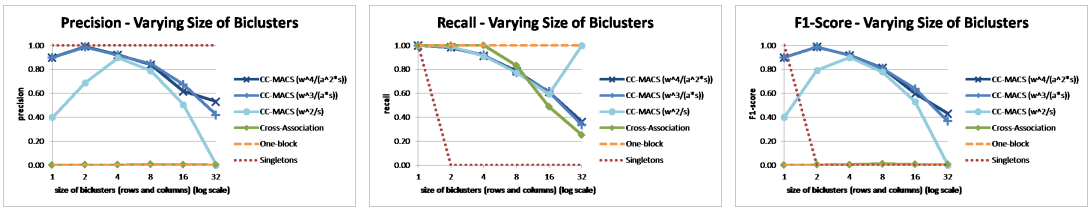
Figure 4.6 shows the results when we vary the number of biclusters. We fix $m = n = 1024$, $r = s = 4$, and $p = 0$, and let $1 \leq k \leq 256$. The Singletons method has perfect precision since there are no false positives, but 0 recall; on the other hand, the One-block method has perfect recall but close to 0 precision. The CC-MACS algorithm with the μ_2 and μ_1 metrics out-perform the other algorithms in finding a good balance between precision and recall, as measured by the F_1-score. Performance inevitably deteriorates as the number of biclusters increases, as they are more likely to overlap and create conflicts in finding a hard co-clustering.

In Figure 4.7, we fix $m = n = 1024$, $k = 16$, and $p = 0$, and vary the size of the biclusters from 1×1 to 32×32. The results are similar to the previous experiment, with performance reaching a peak around 2×2 or 4×4, and declining as the size of the biclusters increases (for



(a) (b) (c)

Figure 4.6: Precision, recall, and F_1-score of several algorithms as the number of biclusters varies.



(a) (b) (c)

Figure 4.7: Precision, recall, and F_1-score of several algorithms as the size of biclusters varies.

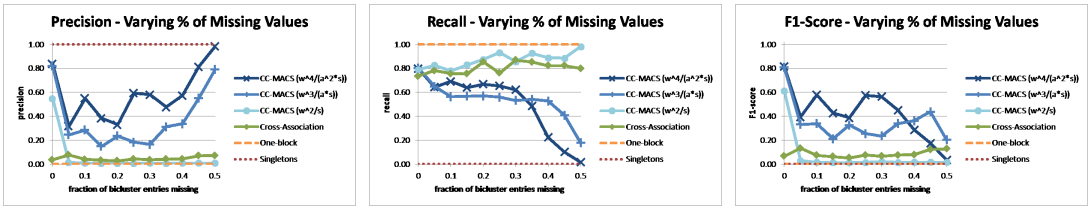
smaller k, the peak would be later).

Figure 4.8 analyzes robustness to missing values. We fix $m = n = 1024$, $k = 8$, and $r = s = 16$, and let $0 \leq p \leq 0.5$. Results show that the CC-MACS algorithm with the μ_2 metric still achieves good precision and recall with up to 30% missing.

4.3.2 Finding Block Structure

We now evaluate the ability of the CC-MACS algorithm to find dense biclusters in real-world matrices with known structure. We looked through the NIST Matrix Market repository, and chose several matrices from the domains of finite element modeling and quantum chemistry because of their clear block structure. Some of the matrices contain complex-valued entries. In the following experiments, we treat all data as $\{0, 1\}$-matrices, where a 1 indicates the presence of a non-zero value.

Figure 4.9 shows the results from the Cross-Association and CC-MACS algorithms. In particular, the CC-MACS algorithm with the μ_2 metric is seen to be effective at identifying large, dense biclusters in these datasets. The μ_0 metric returned the trivial single-block co-clustering on all datasets, reflecting that it does not put enough weight on the density of a



(a) (b) (c)

Figure 4.8: Precision, recall, and F_1-score of several algorithms as the percentage of missing values varies.

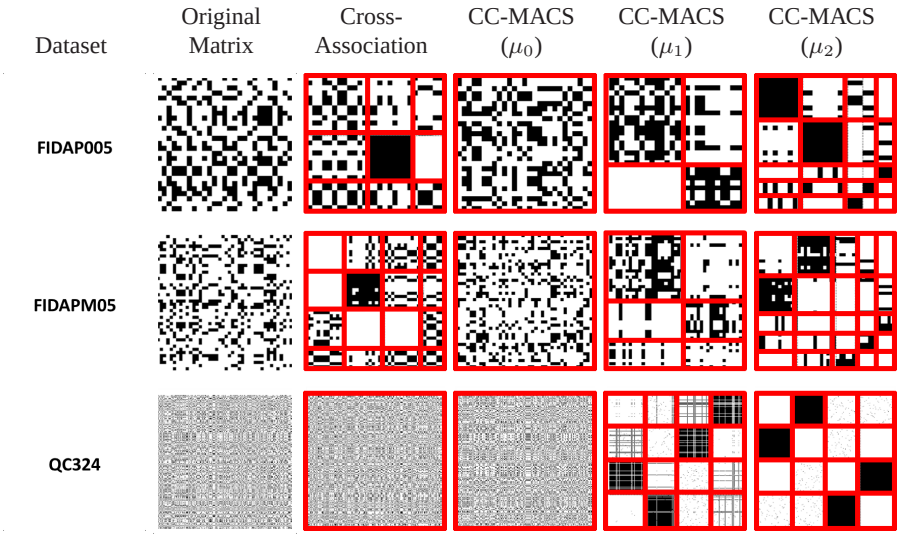


Figure 4.9: Real-world datasets from finite element modeling (FIDAP005 and FIDAPM05) and quantum chemistry (QC324). The red lines indicate co-clusterings found by Cross-Association and CC-MACS algorithms.

bicluster. The μ_1 and μ_2 metrics, on the other hand, seem to have performed quite well – in fact, they achieve a lower encoding cost than the co-clustering output by the Cross-Association algorithm itself. The Cross-Association algorithm may perform poorly on these datasets either because it finds a local optimum before the global optimum is reached, or because in sparse matrices the encoding cost may be minimized by having a single bicluster containing the entire matrix.

4.3.3 Discovering Functional Communities in Social Media

Next we perform experiments on the Meme-Tracker dataset, a large collection of memes extracted from the web by Leskovec et al. [62]. To study the dynamics of the news cycle, they

# of Memes	# of Domains	Density	Topic
26	21	98.2%	St. Jude's Children's Hospital
178	5	96.1%	Brazilian news sites
39	6	98.7%	Spanish news sites
20	6	99.2%	Tech conference and magazine sites
17	6	100.0%	Political blogs

Table 4.4: Top biclusters returned by the CC-MACS algorithm on the Meme-Tracker dataset.

processed text from hundreds of thousands of blogs and news websites to extract distinguishable phrases that suddenly appeared with unusual frequency. They classified these into phrase clusters – each representing a meme – to account for variations in spelling, truncation, and other modifications that may occur as a phrase spreads through the web. From this data, we constructed a binary matrix where each row corresponds to one of the memes, each column corresponds to a web domain, and entry (i, j) is a 1 if and only if the ith meme was mentioned on a website at the jth domain. After filtering out domains with less than 10 memes, the resulting matrix has 71,566 rows; 47,228 columns; and 4,026,266 non-zero entries (0.1% of the matrix). We will refer to this as the MT matrix.

We ran the CC-MACS algorithm on the MT matrix using the μ_0, μ_1, and μ_2 metrics. The μ_0 metric returned the trivial single-block co-clustering, as in the previous experiment, but the μ_1 and μ_2 metrics yielded more illuminating results. Table 4.4 shows some of the top biclusters found.

We see that the CC-MACS algorithm identified several large and very dense biclusters in the MT matrix, and furthermore, that the biclusters correspond to clearly identifiable communities of web domains that participate in disseminating much of the same content.

4.4 Discussion

4.4.1 Extensions and Applications

Our approach to co-clustering may be particularly helpful in *collaborative filtering*, where the goal is to make recommendations for a given user based on feedback from users with similar preferences.

Although in our work we focused on applications that entail a binary relationship, the metrics we propose can be equally applied to real-valued matrices. In Section 4.2.1, we define the density of a bicluster as being dependent on the sum of values and the size of the bicluster, which does not require that the entries be in $\{0, 1\}$. In fact, both of the properties described in Section 4.2.2 still hold for our metrics when the matrix is real-valued.

The CC-MACS algorithm as presented here is a static algorithm. Although the existence of a functional community implies some degree of stability, community structure may change over time. In networks where such changes tend to occur more rapidly, it may be of interest to develop a dynamic version of the CC-MACS algorithm that seamlessly adapts to new data.

Other possible directions for future work include considering a more general class of matrix partitions, not just those formed by a co-clustering of the rows and columns; providing bounds on the approximation factor of our heuristic algorithm; or adapting our approach for distributed computation.

4.4.2 Limitations of Our Approach

The efficiency of the CC-MACS algorithm relies on the metric being of a particular form, the sum over values of a function applied to each bicluster in the co-clustering. For some applications, however, the desired metric may not fit this form. To address this limitation, future work could analyze run-time bounds under various relaxations of this condition.

4.4.3 Significance and Impact

We have presented a new approach for discovering hidden relationships in bipartite data, called the CC-MACS (Co-Clustering via Maximal Anti-Chain Search) algorithm. We first construct k-d trees on the rows and columns using random projections, and then utilize the dual tree structure to efficiently search for a co-clustering which optimizes the value of a given metric. We traverse the trees entirely instead of terminating the algorithm if a local optimum is reached, thus better exploring the breadth of the search space for a globally optimal solution.

The literature on co-clustering spans multiple disciplines, but different applications and data characteristics motivate different approaches. Although the metrics we suggest are motivated

by the task of finding dense biclusters, the CC-MACS algorithm can be used with any metric. While in some real-world scenarios matrices are assumed to have a block diagonal structure, our method does not make that assumption, and therefore can be effective even when that assumption does not hold. Our algorithm is designed to leverage sparsity in the data, running in $O(N \log^2(mn))$ time for sparse matrices. However, we get a further improvement for dense matrices, running in $O(mn \log(mn))$ time. This flexibility to metric, matrix structure, and density makes our algorithm applicable across domains with different goals and data characteristics.

Most related work in the computer science literature assumes that it is desirable to have a small number of clusters, leading to methods that are based on dimensionality reduction, or require the maximum number of row and column clusters to be specified in advance. While there are tasks such as matrix compression where minimizing the number of clusters is essential, we claim that in many real-world applications this is not the case – if the data contains many dense but disjoint biclusters, it may be reasonable to return a co-clustering with hundreds or thousands of row and column clusters. We hope that this observation will motivate others to develop algorithms that also overcome the limitations of the "fewer is better" mentality.

Chapter 5

Modeling Collaboration in Academia

5.1 Introduction

5.1.1 Background and Motivation

In academia, the success of a researcher is often measured by metrics such as number of papers published and how frequently one's work gets cited. These factors play an important role in decisions of tenure, promotions, and awards. However, while each researcher has individual goals, much scientific and academic progress is the result of collaborative efforts. In this work, we seek to better understand the mechanisms driving academic collaboration.

Although collaboration is an essential aspect of most academic disciplines, it has been given relatively little attention in the literature. Most measures of academic impact are based on a researcher's publication record, and pay no regard to collaboration. Others evaluate impact via centrality measures on an aggregated coauthorship graph. However, a researcher's behavioral patterns may change over time. The existing literature that looks at temporal aspects of publication and citation activity tend to assume that the processes driving them are static or have static parameters. We suggest that more sophisticated models are needed to understand the intricacies of collaborative behavior.

Researchers exhibit a wide range of different work habits and behaviors. Some distribute their time among many projects, while others focus on only a few projects at a time. Some engage in mentoring relationships, while others choose to collaborate mostly with their peers. These behaviors may be motivated by a variety of factors such as institutional needs, academic field, stage in career, funding situation, and affinity for teaching. We pose the question: "If researchers were motivated by X, what would the world of academic research look like?" In the current work, we purport to answer this question by first developing a game-theoretic model

of academic collaboration, and then studying the outcome of the game when each researcher is trying to optimize a given objective function.

5.1.2 Related Work

Bibliographic Metrics

Across all academic disciplines, it is natural to want to measure the impact of an individual and his or her work. Consequently, many metrics have been proposed, based on properties of an individual's research output. These start with simple counts like the number of publications (in selective venues) or the total number of citations across all publications, and become progressively more complex. Given the attention such metrics receive, there has been much effort in designing them to be meaningful. For example, total paper counts give little indication of the quality of the work. Aggregate citation counts are distorted by a single highly-cited paper, and so do not indicate the breadth of the researcher's work.

In 2005, Hirsch proposed the h-index: the largest integer h such that the author has published at least h papers with at least h citations each [42]. This measure has an intuitive appeal, and is not unduly influenced by a single high-impact paper, nor by a multitude of low-impact publications. Since then, a plethora of variations and alternative indices have been proposed to address perceived shortcomings of the h-index [29, 48]. Most of these measures evaluate an author solely based on his or her individual publication record. However, modern scientific research tends to be highly collaborative in nature.

There has been some effort in recent years to design bibliographic metrics that take collaboration into account. Abbasi et al. proposed an index that rewards an author for collaborating with top researchers [2]. Kameshwaran et al. defined a metric combining strength of publication record with eigenvector centrality to identify prominent researchers in the collaboration network [51]. In previous work, we proposed the Social h-index, which attributes partial credit for a researcher's success to the coauthors whose joint work contributed to that success [22].

Temporal Models

Several models have been proposed to study how publications and citations accumulate over time. In his original paper, Hirsch suggests a model in which a researcher publishes a constant number of papers per year, and each paper accumulates a constant number of additional citations per year [42]. This results in linear growth of the h-index, with the earliest papers accumulating the most citations.

Other works have further studied how the h-index of a researcher grows over time, through simulation models or empirical studies. Wu et al. track the h-indices of 47 Nobel Prize winners as functions parameterized by time and categorize them into five shapes: linear, convex, concave, S-shaped, and IS-shaped [97]. They also examine the "freshness" of the papers contributing to the h-index – whether they were published early or late in the researcher's career – and find that a researcher's best papers tend to be distributed throughout her career.

Guns and Rousseau look at how citations of a paper accumulate over time [39]. They suggest a peak-decay model, in which the number of citations a given paper receives increases each year until a peak year and then decreases. Note that under this model, the total number of citations a paper receives is usually bounded (but could differ by paper), whereas under Hirsch's model, papers accumulate citations unboundedly. They show through simulations that by varying the parameters (peak year, height of peak, and rate of decay) – or choosing them stochastically – growth of the h-index under the peak-decay model can be linear, concave, or S-shaped.

Cardillo et al. empirically study the correlation between stability of local graph structure over time and the willingness of individuals to compromise their own interests in favor of social cooperation [15], but stop short of suggesting a mechanism that would explain such behavior.

5.1.3 Contributions and Outline

In this work, we aim to understand the mechanisms underlying academic collaboration. Using tools from the field of Game Theory, we study how collaboration may arise as the result of interplay between reseachers' individually-motivated behaviors.

Notation	Description
$cit(p)$	the total # of citations received by paper p
$cit_y(p)$	the # of citations received by paper p in year y
$A(p)$	the set of authors of paper p
$P(a)$	the set of papers authored by a
$P_y(a)$	the set of papers authored by a in year y
$\chi_y(a)$	the citation profile of researcher a in year y
$h_y(a)$	the h-index of researcher a in year y
$H_y(a)$	the h-profile of researcher a in year y
$\widetilde{H}_y(a)$	the h-augmenting profile of researcher a in year y

Table 5.1: Table of basic notation

We begin by building a theoretical model for how researchers collaborate and how collaboration affects the number of citations a paper receives, supported by observations from a large real-world publication and citation dataset. Using this model, we study researchers' collaborative behavior over time under the premise that each person wants to maximize his or her academic success in terms of both the quality and quantity of her research output.

Our main contributions can be summarized as follows:

- A game-theoretic framework modeling academic collaboration as a repeated game
- Formal analysis of collaboration strategies and game equilibria

5.2 Methodology

We first introduce some basic terminology and notation. Using a game-theoretic framework, we then describe a game of academic collaboration with which we can simulate researchers' collaborative behavior over time.

5.2.1 Preliminaries

We begin with some definitions, including a more general definition of the h-index, originally proposed in [42]. A summary of notation is provided in Table 5.1.

We define the *citation profile* of a set of papers P, denoted $\chi(P)$, to be the multi-set $\{cit(p) : p \in P\}$; and the citation profile of a researcher a to be $\chi(a) = \chi(P(a))$. When multiple years are being considered, we use $\chi_y(a)$ to denote the citation profile of researcher a in year y.

We define the *h-index* of a multi-set of non-negative integers Z, denoted $h(Z)$, to be the largest integer h such that at least h elements of Z are greater than or equal to h:

$$h(Z) = \max\left\{h : |\{z \in Z,\ z \geq h\}| \geq h\right\}.$$

For simplicity of notation, we define the h-index of a set of papers P to be $h(P) = h(\chi(P))$; and the h-index of a researcher a to be $h(a) = h(P(a)) = h(\chi(P(a)))$. When multiple years are being considered, we use $h_y(a)$ to denote the h-index of researcher a in year y.

We define the *h-profile* of a multi-set of non-negative integers Z, denoted $H(Z)$, to be the multi-set of integers in Z that are greater than or equal to $h(Z)$:

$$H(P) = \{z \in Z : z \geq h(Z)\}.$$

We similarly define the h-profile of a set of papers P to be $H(P) = H(\chi(P))$; and the h-profile of a researcher a to be $H(a) = H(P(a)) = H(\chi(P(a)))$. When multiple years are being considered, we use $H_y(a)$ to denote the h-profile of researcher a in year y.

Sometimes we are only interested in the papers with strictly more than h citations. We define the *h-augmenting profile* of a multi-set of non-negative integers Z, denoted $\widetilde{H}(Z)$, to be the multi-set of integers in Z that are strictly greater than $h(Z)$:

$$\widetilde{H}(P) = \{z \in Z : z > h(Z)\}.$$

We similarly define the h-augmenting profile of a set of papers P to be $\widetilde{H}(P) = \widetilde{H}(\chi(P))$; and the h-augmenting profile of a researcher a to be $\widetilde{H}(a) = \widetilde{H}(P(a)) = \widetilde{H}(\chi(P(a)))$. When multiple years are being considered, we use $\widetilde{H}_y(a)$ to denote the h-augmenting profile of researcher a in year y. Intuitively, the h-augmenting profile indicates progress towards increasing the h-index.

Let Z and Z' be multi-sets of non-negative integers. We say Z is *weakly h-preferable* to Z', denoted $Z \succeq_h Z'$, if $h(Z) \geq h(Z')$ and $(\forall\ z_0 > h(Z))\ |\{z \in Z : z \geq z_0\}| \geq |\{z \in Z' : z \geq z_0\}|$. We say Z is *strongly h-preferable* to Z', denoted $Z \succ_h Z'$, if in addition either $h(Z) > h(Z')$ or $\exists\, z_0 > h(Z)$ for which the inequality is strict. When P and P' are two sets of papers, we write $P \succeq_h P'$ to denote that $\chi(P) \succeq_h \chi(P')$, and $P \succ_h P'$ to denote that $\chi(P) \succ_h \chi(P')$.

Next, we propose a model with which to study academic collaboration over time.

5.2.2 Game-Theoretic Model

To model the collaborative behavior of researchers in academia, we appeal to field of Game Theory. We consider a model where in year y, each researcher a has a fixed amount of *research potential* $Q_y(a)$ to be invested in writing papers, and the total number of citations that a paper receives reflects the amount of research potential that was invested in the paper by its authors. For simplicity of analysis, we model all citations as being received in the same year that the paper is published. We also suggest that there is a practical limit on the number of coauthors that can meaningfully contribute to a paper, and in the following analysis limit a paper to two coauthors. Future work could revisit the analysis under a more realistic or general model.

A *game* is a way of modeling the decisions of a set of rational *players* whose *actions* collectively determine an *outcome*. A player's goal is to achieve an outcome of maximal *utility* to that player. We model collaboration in academia as a *repeated game*, where the same base game is played multiple times, and in each iteration players choose actions simultaneously.

We formalize a repeated game played by a set of researchers, explicitly defining the actions available to each researcher in each year, the outcomes determined by those actions, and the utility of each possible outcome to each researcher. We refer to this as the Academic Collaboration (AC) game:

- **Players:** Let A be a set of researchers, each $a \in A$ initially having published a set of papers resulting in citation profile $\chi_0(a)$.

- **Actions:** In year y, each researcher $a \in A$ has $Q_y(a)$ units of research potential to distribute amongst individual and collaborative projects. Formally, a constructs a finite sequence of non-negative integers $\mathbf{q}_\mathbf{y}^\mathbf{a}$, and for each potential coauthor $a' \in A$ a sequence $\mathbf{q}_\mathbf{y}^{\mathbf{a},\mathbf{a}'}$, such that
$$\sum_i \mathbf{q}_\mathbf{y}^\mathbf{a}[i] + \sum_{a'} \sum_i \mathbf{q}_\mathbf{y}^{\mathbf{a},\mathbf{a}'}[i] = Q_y(a).$$

- **Outcome:** In year y, a paper is produced for each project, which receives citations commensurate with the research potential invested by its coauthors. A researcher a becomes a coauthor on a paper p by investing a non-zero amount of research potential $q(a,p)$ in it. Formally: Let cit be the citation function, which maps a non-empty

multi-set $\{q(a,p)\}_{a\in A'}$ consisting of the research potential invested in a project by its coauthors $A' \subseteq A$ to the number of citations the resulting paper p will receive. For $i > |\mathbf{q}|$, define $\mathbf{q}[i] = 0$. For all a, i such that $\mathbf{q}^{\mathbf{a}}_{\mathbf{y}}[i] > 0$, a paper will be published which will receive $\mathrm{cit}(\{\mathbf{q}^{\mathbf{a}}_{\mathbf{y}}[i]\})$ citations, singly-authored by a. For all $\{a, a'\} \in \binom{A}{2}$ and i such that $\mathbf{q}^{\mathbf{a},\mathbf{a}'}_{\mathbf{y}}[i] + \mathbf{q}^{\mathbf{a}',\mathbf{a}}_{\mathbf{y}}[i] > 0$, a paper will be published which will receive $\mathrm{cit}\left(\left\{\mathbf{q}^{\mathbf{a},\mathbf{a}'}_{\mathbf{y}}[i], \mathbf{q}^{\mathbf{a}',\mathbf{a}}_{\mathbf{y}}[i]\right\}\right)$ citations, for which a (resp. a') is a coauthor if and only if $\mathbf{q}^{\mathbf{a},\mathbf{a}'}_{\mathbf{y}}[i] > 0$ (resp. $\mathbf{q}^{\mathbf{a}',\mathbf{a}}_{\mathbf{y}}[i] > 0$).

- **Utility:** The function $Util_y(a) = h_y(a)$ indicates the utility for researcher a at the end of year y.

We will consider the AC game of *infinite horizon*, which means that each player wants to maximize his utility in the limit, rather than after some pre-specified number of years.[1] The Game Theory literature considers several ways to compare player preferences in infinite games. Our approach is most similar to the overtaking criterion presented in [85].

The *game state* represents, at any point in the game, all information that may help determine the available actions, corresponding outcomes, and utilities of the players. In the AC game, we define the game state to consist of the citation profiles of the researchers.

A *strategy* is a set of rules that govern which action a player will take given her knowledge of the game state. In the current work, we only consider deterministic strategies.

Let s be a set of strategies for a game, one per player; this is referred to as a *strategy profile*. For the purpose of analysis, we take two strategy profiles to be equal if they always result in the same outcome. When considering multiple strategy profiles, we denote by $P^s_y(a)$, $\chi^s_y(a)$, $h^s_y(a)$, $H^s_y(a)$, $\widetilde{H}^s_y(a)$, and $Util^s_y(a)$ the papers, citation profile, h-index, h-profile, h-augmenting profile, and utility, respectively, for player a after y iterations of the game when the players follow their respective strategies in s; and by $W^s(A)$ the social welfare under s. We denote by s_a the strategy for player $a \in A$ under strategy profile s,[2] and by $s_{\bar{a}}$ the strategies for

[1] Although in reality a researcher lives for only a finite number of years, infinite games are arguably a reasonable model of human behavior when "players examine a long-term situation without assigning a specific status to the end of the world" [86].

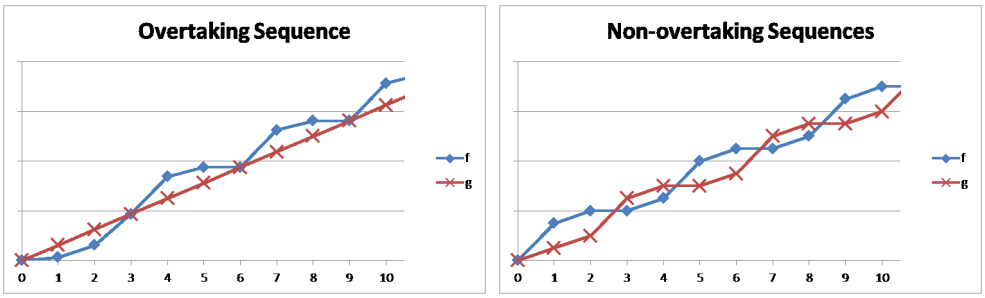
[2] For convenience, we also use s_a to denote the singleton set containing that strategy; in each use case, the meaning should be clear from context.

all players other than a; by $s_{A'}$ the strategies for players in $A' \subseteq A$, and by $s_{\bar{A}'}$ the strategies for players not in A'.

Let f_n and g_n be two infinite real-valued sequences. We say that f_n *overtakes* g_n if $\limsup_{n\to\infty} f_n - g_n > 0$ and $\liminf_{n\to\infty} f_n - g_n \geq 0$.[3] We note that there are three (mutually exclusive and exhaustive) possibilities:

- f_n overtakes g_n
- g_n overtakes f_n
- neither f_n nor g_n overtakes the other

These are illustrated in Figure 5.1.



(a) (b)

Figure 5.1: (a) Sequence f_n overtakes g_n. (b) Neither sequence f_n nor g_n overtakes the other.

Multiple notions of equilibrium have been proposed in the literature. Due to the collaborative nature of the AC game, we consider a set of strategies to be in equilibrium if no two researchers would prefer to deviate from their current strategies in order to collaborate with one another. We formalize this by generalizing the notion of stability presented in [32].

Given a strategy profile s for the players in an infinite game, we say that the subset of players $A' \subseteq A$ is *unstable under* s if there exist alternate strategies $s'_{A'}$ for the players in A' such that $(\forall\ a \in A')\ Util_n^{s_{\bar{A}'} \cup s'_{A'}}(a)$ overtakes $Util_n^s(a)$. We define a strategy profile s^* to be a *k-stable equilibrium* if there does not exist an unstable set of size at most k. Throughout the rest of this chapter, we use the term *equilibrium* to refer to a 2-stable equilibrium.

In the next section, we use the AC game to examine how researchers' individually-motivated

[3] In [85], f_n overtakes g_n if $\liminf_{n\to\infty} f_n - g_n > 0$. Our definition is more inclusive, additionally allowing for the situation in Figure 5.1(a).

behavior can lead to academic collaboration.

5.3 Evaluation

First, we build a model of academic collaboration based on data extracted from a large corpus of Computer Science publications. In particular, we analyze how researchers split their effort between multiple papers, and the relationship between the authors of a paper and the number of citations it receives. Next, we explore the single-player, two-player, and multi-player versions of the AC game. For each version, we analyze the asymptotic behavior and equilibria when each player is trying to maximize his or her h-index.

5.3.1 Collaboration Model

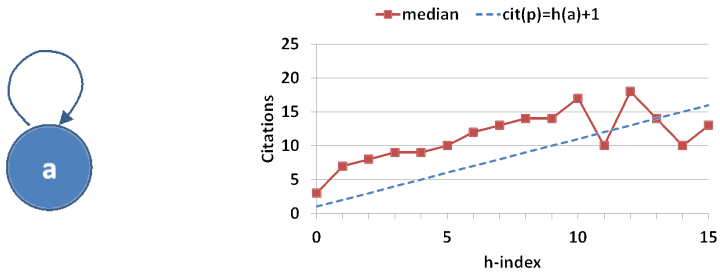
In Section 5.2.2, we proposed a game-theoretic model of academic collaboration in which in year y, each researcher a has a fixed amount of research potential $Q_y(a)$ to be invested in writing papers, and the total number of citations that a paper receives reflects the amount of research potential that was invested in the paper by its authors. We now further specify this mechanism by analyzing publication and citation data from the field of Computer Science. We extract all publications, along with authors and number of citations received, from a snapshot of the DBLP database, which contains approximately 1 million researchers and 2 million publications.

We first analyze the simple case of a paper published by a single author who had no other publications that same year,[4] and explore the relationship between the number of citations a paper receives and several attributes of the author: number of papers published previously, total number of citations received previously, and current h-index. We compute Spearman's rank correlation coefficient for each of the attributes,[5] and find that the h-index has the highest correlation with a value of 0.34, compared to paper count with a value of 0.28 and citation sum with a value of 0.08. Therefore, in subsequent analysis, we use the h-index as a proxy for the research potential of an author.

---

[4]The assumption is that if a researcher published only one paper in a given year, then all of her effort went into that paper. In reality, she could have worked on projects that were not published that year, but that is hard to evaluate empirically since unpublished papers are not captured in the data.

[5]We choose this over the more common Pearson's coefficient because it is more robust to non-linear relationships.



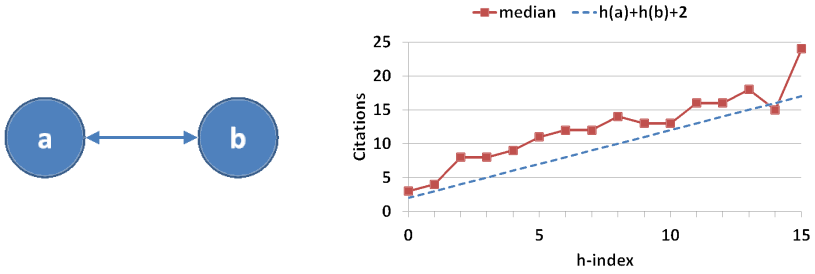
(a) (b)

Figure 5.2: The h-index of the author versus the median number of citations received across all single-authored papers for which the author published no other papers the same year.

In Figure 5.2, we take a closer look at the relationship between the h-index of the author and the number of citations a paper receives. The plot shows the median number of citations received on papers singly-authored by a researcher with h-index h for each value of h. We use the median because there are a few extreme outliers which skew the average to the right, and we are looking for a model which represents a typical researcher. Comparison to the best-fit line demonstrates visually that the two quantities have a linear relationship up until an h-index of about 10, indicating that the number of citations a single-authored paper receives is proportional to the h-index of the author when he puts all of his effort into the paper. For values of $h > 10$, the fluctuation may be a result of high variance and too few data points.

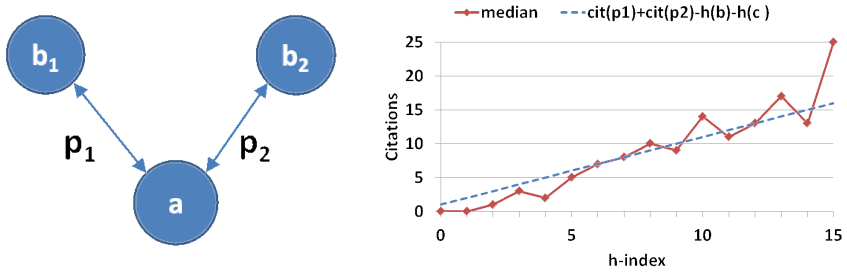
Next, we look at the case of papers with multiple authors. To isolate this aspect of the model, we consider two-author papers where neither of the authors published any other papers in the same year, as illustrated in Figure 5.3(a). In Figure 5.3(b), we plot the sum of the h-indices of the authors versus the median number of citations received across all such papers. We again observe a linear relationship, indicating that the combined research potential of multiple authors is additive when they put all of their effort into the paper.

Finally, we investigate what happens when an author publishes multiple papers in the same year by narrowing our focus to instances where aside from the author of interest, none of the



(a) (b)

Figure 5.3: The sum of the h-indices of the coauthors versus the median number of citations received across all two-authored papers where neither author published any other papers in the same year. The dashed line is the number of citations predicted by our model.



(a) (b)

Figure 5.4: The h-index of an author a who published two papers p_1 and p_2 in the same year with coauthors b_1 and b_2 respectively, neither of which published any other papers in the same year, versus the median of $(cit(p_1) - h(b_1)) + (cit(p_2) - h(b_2))$.

coauthors published any other papers in the same year. This scenario is illustrated in Figure 5.4(a). Using the previous result of research potential being additive across multiple coauthors, we plot the h-index of the author of interest against the value

$$\sum_{p \in P_y(a)} \left(cit(p) - \sum_{b \in A(p) \setminus \{a\}} h(b) \right)$$

in Figure 5.4(b). The plot shows a linear relationship, indicating that the distribution of an author's research potential across multiple papers is also linear.

Based on the observations above, we formalize our model with the following three properties:

1. In year y, a researcher a has $Q_y(a) = h_y(a) + 1$ units of research potential to be invested in writing papers.

2. An individual's research potential can be distributed amongst any number of papers to be published in that year:
$$Q_y(a) = \sum_{p \in P_y(a)} q(a,p),$$
where $q(a,p)$ is the amount of research potential invested by researcher a in paper p.

3. A paper p will receive in total $cit(p) = \text{cit}\left(\{q(a,p)\}_{a \in A(p)}\right) = \sum_{a \in A(p)} q(a,p)$ citations.

We now use this model to study the AC game.

5.3.2 Single-Player Game

First, we consider the AC game when there is only one player, researcher a. In this case, a may only write single-author papers; the question is how many papers to write and how to optimally distribute her research potential between them.

We begin by analyzing how the utility function grows when a puts all of her effort into writing a single paper each year.

Proposition 5.1. *Consider the single-player AC game of infinite horizon. Let s^* denote the strategy profile where each year the player a invests all research potential into a single paper. Then the limit behavior for player a's utility under s^* is*

$$\limsup_{n \to \infty} \; Util_n^{s^*}(a) \sim \sqrt{2n}.$$

Proof. If the claim holds for $h_0(a) = 0$, then it also holds for $h_0(a) > 0$, so assume that $h_0(a) = 0$. Following strategy s^*, from the time a reaches an h-index of h', it will take $h' + 1$ years to accumulate $h' + 1$ papers with $h' + 1$ citations each. Thus a requires a total of $n = \sum_{i=1}^{h} i = \frac{h(h+1)}{2}$ years to achieve an h-index of h. Conversely, as the number of years n goes to infinity, a achieves a utility of

$$\limsup_{n \to \infty} \; Util_n^{s^*}(a) = \limsup_{n \to \infty} \; h_n^{s^*}(a) = \limsup_{n \to \infty} \; \left\lfloor \frac{-1 + \sqrt{1 + 8n}}{2} \right\rfloor \sim \sqrt{2n}. \qquad \square$$

We now compare a's success under the single-paper strategy relative to other possible ways of distributing her effort.

Lemma 5.2. *Consider the single-player AC game of infinite horizon. Let s^* denote the strategy profile where each year the player a invests all research potential into a single paper. Then for all strategy profiles $s \neq s^*$, $h_n^{s^*}(a)$ overtakes $h_n^s(a)$.*

Proof. Consider a strategy profile $s \neq s^*$. Let y^* be the first year in which the outcome is different under s^* and s, so that $h_{y^*-1}(a) = h_{y^*-1}^{s^*}(a) = h_{y^*-1}^s(a)$ and $\widetilde{H}_{y^*-1}(a) = \widetilde{H}_{y^*-1}^{s^*}(a) = \widetilde{H}_{y^*-1}^s(a)$. Since s^* produces a single paper that will receive $Q_{y^*}(a) = h_{y^*-1}(a) + 1$ citations, a's strategy under s must split the research potential between at least two papers, each therefore receiving at most $h_{y^*-1}(a)$ citations, resulting in $H_{y^*}^{s^*}(a) \succ_h H_{y^*}^s(a)$. It follows by induction that $H_y^{s^*}(a) \succ_h H_y^s(a)$ for all $y \geq y^*$, and furthermore, that $h_y^{s^*}(a) > h_y^s(a)$ for all years $y \geq y^*$ in which $h^{s^*}(a)$ increases. By definition, $h_n^{s^*}(a)$ overtakes $h_n^s(a)$. □

Theorem 5.3. *Consider the single-player AC game of infinite horizon. Let s^* denote the strategy profile where each year the player a invests all research potential into a single paper. Then s^* is the only equilibrium.*

Proof. This follows directly from Lemma 5.2. □

We have shown the strategy described above to be optimal for the single-player AC game. However, a researcher may hope to have a greater impact by collaborating with others. We explore this possibility in the following sections.

5.3.3 Two-Player Game

We now consider the AC game with two players, a and a'. For simplicity, we only analyze the case where $H_0(a) = H_0(a')$, i.e. initially both researchers have the same h-profile; the results can be generalized for arbitrary initial citation profiles. Note that if all papers produced through year y are joint between a and a', then $h_y(a) = h_y(a')$, $H_y(a) = H_y(a')$, and $\widetilde{H}_y(a) = \widetilde{H}_y(a')$, in which case we will denote them by h_y, H_y, and $\widetilde{H}_y$, respectively.

We begin by considering two collaborative strategy profiles: one where both players pool all their effort into a single joint paper, and another where they collaborate on two papers simultaneously. We analyze how the players' utility functions grow under each scenario.

Proposition 5.4. *Consider the two-player AC game of infinite horizon, where $H_0(a) = H_0(a')$. Let s^* denote the strategy profile where each year the players invest their research potential into a single joint paper. Then the limit behavior for each player's utility under s^* is*

$$\limsup_{n\to\infty} \ Util_n^{s^*} \geq \frac{n}{2}.$$

Proof. If the claim holds for $h_0 = 0$, then it also holds for $h_0 > 0$, so assume that $h_0 = 0$. We use recursion to give a bound on $y_h^{s^*}$, the number of years needed to achieve an h-index of h under s^*. We have that $y_0^{s^*} = 0$, and $y_h^{s^*} \leq y_{\lceil h/2 \rceil - 1}^{s^*} + h$, since after they have achieved h-index of $\lceil h/2 \rceil - 1$, each of the following h years they will produce a paper with at least h citations each. We get the following bound:

$$\begin{aligned} y_h^{s^*} &\leq y_{\lceil h/2 \rceil - 1}^{s^*} + h \\ &\leq y_{\lfloor h/2 \rfloor}^{s^*} + h \\ &\leq h + \frac{h}{2} + \frac{h}{4} + \dots \\ &\leq 2h \end{aligned}$$

Conversely, $h \geq y_h^{s^*}/2$, so as the number of years n goes to infinity, each player achieves a utility of

$$\limsup_{n\to\infty} \ Util_n^{s^*} = \limsup_{n\to\infty} \ h_n^{s^*} \geq \frac{n}{2}. \qquad \square$$

Proposition 5.5. *Consider the two-player AC game of infinite horizon, where $H_0(a) = H_0(a')$. Let $s^{\vDash}$ denote the strategy profile where each year the players split their research potential evenly between two joint papers. Then the limit behavior for each player's utility under $s^{\vDash}$ is*

$$\limsup_{n\to\infty} \ Util_n^{s^*} \sim 2\sqrt{n}.$$

Proof. If the claim holds for $h_0 = 0$, then it also holds for arbitrary initial citation profiles, so assume that $h_0 = 0$. Following strategy $s^{\vDash}$, from the time the players each reach an h-index of h', it will take $\lceil (h'+1)/2 \rceil$ years to accumulate $h'+1$ papers with $h'+1$ citations each. Thus a total of $n = \sum_{i=1}^{h} \lceil i/2 \rceil \geq \frac{h(h+1)}{4}$ years are required to achieve an h-index of h. Conversely, as the number of years n goes to infinity, each player achieves a utility of

$$\limsup_{n\to\infty} \ Util_n^{s^{\vDash}} = \limsup_{n\to\infty} \ h_n^{s^{\vDash}} = \limsup_{n\to\infty} \ \left\lfloor \frac{-1 + \sqrt{1+16n}}{2} \right\rfloor \sim 2\sqrt{n}.$$

$\square$

We now examine how these two strategy profiles compare to other possible strategies for the two-player game.

Lemma 5.6. *Consider the two-player AC game of infinite horizon, where $H_0(a) = H_0(a')$. Let s^* denote the strategy profile where each year the players invest their research potential into a single joint paper, and let $s^{\vDash}$ denote the strategy profile where each year the players split their research potential evenly between two joint papers. Let $S^{\{*,\vDash\}}$ denote the set of strategy profiles that each year prescribe either s^* or $s^{\vDash}$. Then for any strategy profile $s \notin S^{\{*,\vDash\}}$, $\exists\, s' \in S^{\{*,\vDash\}}$ such that $h_n^{s'}$ overtakes both $h_n^s(a)$ and $h_n^s(a')$.*

Proof. Consider a strategy profile $s \notin S^{\{*,\vDash\}}$. Consider the strategy profile s' which is identical to s for game states in which s prescribes actions according to s^* or $s^{\vDash}$, and behaves like s^* otherwise. Let y' be the first year in which s and s' differ, so that $H_{y'-1} = H^{s'}_{y'-1} = H^s_{y'-1}$. Let $P^s_{y'}$ denote the set of papers produced by s in year y', then we have $\sum_{p \in P^s_{y'}} cit(p) = 2(h_{y'-1}+1)$. Since s differs from $s^{\vDash}$ in year y', there can be at most one paper with $\geq h_{y'-1}+1$ citations; and since it differs from s^*, no paper can have $2(h_{y'-1}+1)$ citations; it follows that $H^{s'}_{y'} \succ_h H^s_{y'}(a)$ and $H^{s'}_{y'} \succ_h H^s_{y'}(a')$. It follows by induction that $H^{s'}_y \succ_h H^s_y(a)$ and $H^{s'}_y \succ_h H^s_y(a')$ for all $y \geq y'$, and furthermore, that $h^{s'}_y > h^s_y(a)$ and $h^{s'}_y > h^s_y(a')$ for all years $y \geq y'$ in which $h^{s'}_y$ increases. By definition, $h^{s'}_n$ overtakes both $h^s_n(a)$ and $h^s_n(a')$. □

Lemma 5.7. *Consider the two-player AC game of infinite horizon, where $H_0(a) = H_0(a')$. Let s^* denote the strategy profile where each year the players invest their research potential into a single joint paper. Then there does not exist a strategy profile $s \neq s^*$ such that either $h^s_n(a)$ or $h^s_n(a')$ overtakes $h^{s^*}_n$.*

Proof. Consider a strategy profile $s \neq s^*$. Let $s^{\vDash}$ denote the strategy profile where each year the players split their research potential evenly between two joint papers, and let $S^{\{*,\vDash\}}$ denote the set of strategy profiles that each year prescribe either s^* or $s^{\vDash}$. If $s \notin S^{\{*,\vDash\}}$ then we are done by Lemma 5.6, so assume $s \in S^{\{*,\vDash\}}$. Let $y^{s^*}_i$ denote the first year such that $h^{s^*}_{y_i} \geq i$; let y^s_i denote the first year such that $h^s_{y_i} \geq i$; and let k_i denote the number of years $y^s_{i-1} \leq y < y^s_i$ in which s differs from s^*. It follows by induction that $y^{s^*}_i - y^s_i \leq k_i - \sum_{j<i} k_j$. In particular, if $k_i < \sum_{j<i} k_j$, then $y^{s^*}_i < y^s_i$, which implies that in year $y^{s^*}_i$ we have $h^{s^*} > h^s$. Since the

sequence $z_0 = 0,\ z_i = \sum_{j<i} z_j$ grows exponentially yet k_i can grow at most linearly, this is guaranteed to happen an infinite number of times. Since h only takes integral values, we have that $\liminf_{n\to\infty} h_n^s - h_n^{s^*} < 0$, and so by definition h_n^s does not overtake $h_n^{s^*}$. □

Theorem 5.8. *Consider the two-player AC game of infinite horizon, where $H_0(a) = H_0(a')$. Let s^* denote the strategy profile where each year the players invest their research potential into a single joint paper, and let $s^{\vDash}$ denote the strategy profile where each year the players split their research potential evenly between two joint papers. Let $S^{\{*,\vDash\}}$ denote the set of strategy profiles that each year prescribe either s^* or $s^{\vDash}$. Then we have the following:*

(a) All equilibria must be in $S^{\{*,\vDash\}}$.

(b) The strategy profile s^* is an equilibrium.

(c) Not all strategy profiles in $S^{\{*,\vDash\}}$ are equilibria.

Proof. Claims (a) and (b) follow directly from Lemmas 5.6 and 5.7, respectively. For claim (c), it is sufficient to show that $s^{\vDash}$ is not an equilibrium, which follows from Propositions 5.4 and 5.4 since the players would rather play according to s^*. □

5.3.4 Multi-Player Game

We now look at the AC game with an arbitrary number of players, A. For simplicity, we only analyze the case where $(\forall\ a \in A)\ H_0(a) = H_0$, i.e. initially all researchers have the same h-profile; the results can be generalized for arbitrary initial citation profiles.

We consider two variants: the "static" AC game, where each player follows the same collaboration strategy each year; and the "dynamic" AC game, where new collaborations may be formed and the distribution of research potential may change.

We represent the static game as a directed graph, each edge (a, a') labeled with a vector $\hat{\mathbf{q}}_{\mathbf{y}}^{\mathbf{a}',\mathbf{a}}$ such that

- $(\forall\ a, a' \in A,\ i \in \mathbb{N}) \quad \hat{\mathbf{q}}_{\mathbf{y}}^{\mathbf{a},\mathbf{a}'}[i] \leq 1;$ and
- $(\forall\ a \in A) \quad \sum_{i\in\mathbb{N}} \hat{\mathbf{q}}_{\mathbf{y}}^{\mathbf{a}}[i] + \sum_{a'\neq a} \sum_{i\in\mathbb{N}} \hat{\mathbf{q}}_{\mathbf{y}}^{\mathbf{a},\mathbf{a}'}[i] = 1.$

That is, the graph dictates what fraction of a player's research potential is invested in each collaboration every year.

Theorem 5.9. *Consider the static multi-player AC game of infinite horizon, where we have that* $(\forall\ a \in A)\ H_0(a) = H_0$. *Let* S^* *be the set of strategy profiles corresponding to perfect matchings on* A*, where each year every pair of players in the matching invests their research potential into a single joint paper.*[6] *Then all of the strategy profiles in* S^* *are equilibria.*

Proof. Consider a strategy profile $s^* \in S^*$. It is obvious that no player can improve her utility if all other players' strategies remain the same, since joint papers are not possible without cooperation from both players. Consider any strategy profile s' differing from s^* only in the strategies of players a_1 and a_2, so that under s' both a_1 and a_2 invest a non-zero fraction of their research potential into a joint paper. By an argument similar to that in the proof of Lemma 5.7, it is not possible that both $h_n^{s'}(a_1)$ overtakes $h_n^{s^*}(a_1)$ and $h_n^{s'}(a_2)$ overtakes $h_n^{s^*}(a_2)$, so by definition a_1 and a_2 do not form an unstable set. Since this is true for all pairs of players, there does not exist an unstable set of at most two players under s^*. Thus s^* is an equilibrium. □

Next, we consider the same strategy profiles in the dynamic setting, with a very different result.

Theorem 5.10. *Consider the dynamic multi-player AC game of infinite horizon, where we have that* $(\forall\ a \in A)\ H_0(a) = H_0$. *Let* S^* *be the set of strategy profiles corresponding to perfect matchings on* A*, where each year every pair of players in the matching invests their research potential into a single joint paper. Then for* $|A| > 2$*, none of the strategy profiles in* S^* *are equilibria.*

Proof. Consider a strategy profile $s^* \in S^*$. Let a_1 and a_2 be two players who are not paired up in the matching, and let a_1' and a_2' be their matched pairs, respectively. We construct a strategy profile s' as follows: All players besides a_1 and a_2 follow their respective strategies under s^*. In years 1 and 2, a_1 and a_2 follow their strategies under s^*; in years 3 and 7, they invest one unit of research potential in a joint paper with a_1' and a_2', respectively, and the rest in a single joint paper between themselves; and in all other years a_1 and a_2 invest all of their research potential in a single joint paper between themselves. It can be shown that $h_n^{s'}(a_1)$ overtakes $h_n^{s^*}(a_1)$ and $h_n^{s'}(a_2)$ overtakes $h_n^{s^*}(a_2)$. Therefore, s^* is not an equilibrium. □

[6] Note that this set is empty when $|A|$ is odd.

We conclude by posing the following open question:

Question. *Does there exist an equilibrium for the dynamic multi-player AC game of infinite horizon?*

5.4 Discussion

5.4.1 Extensions and Applications

Many modifications and extensions to our model are possible. For example, instead of all decisions being deterministic, one could allow for mixed strategies, i.e. where a player's action each year is selected from a probability distribution over possible strategies. Also, under our proposed model, the number of citations received by a paper is determined by the h-indices of the coauthors; alternative models could have a person's research potential be dependent on other variables. A further extension could allow more than two coauthors on a paper, perhaps with a sublinear aggregation function to avoid the degenerate solution of all researchers collaborating on a single giant paper. An even more realistic model could allow the set of authors to change, for example as new researchers enter academia.

Our analysis was performed under the premise that each researcher wants to maximize his or her h-index. However, in real life researchers are motivated by a variety of factors. In previous work, we introduced an alternative to the h-index, the Social h-index, that aims to capture not only the direct impact of a researcher on the research corpus, but also on his or her fellow researchers [22]. Here we consider two variants, the Instantaneous Social h-index and the Progressive Social h-index.

Taking the h-index as a suitable metric for the impact of a person's individual research contributions, we define the *Social h-index* of author a to be

$$\sum_{p \in P(a)} \frac{1}{|A(p)|} \sum_{a' \in A(p)} \mathrm{contrib}(p, a'),$$

where $\mathrm{contrib}(p, a')$ measures how much paper p has contributed to the h-index of a'. That is, for each paper a has coauthored, he gets partial credit for the contribution of that paper to each coauthors' h-index, including himself. We suggest two instantiations of the $\mathrm{contrib}$ function,

one based on whether the paper currently contributes to a designated author's h-index, and the other which considers its cumulative contribution over time.

For the *Instantaneous Social h-index* of author a, denoted $\text{Inst-Soc}^h(a)$, we define

$$\text{contrib}(p,a) = \begin{cases} \dfrac{h(a)}{|H(a)|} & \text{if } p \in H(a) \\ 0 & \text{otherwise} \end{cases}$$

A natural first thought would have been to set $\text{contrib}(p,a) = 1$ if $p \in H(a)$ and 0 otherwise. However, due to ties, we may have $|H(a)| > h(a)$. The above definition maintains the property that $\sum_{p \in P(a)} \text{contrib}(p,a) = h(a)$, even in the case of ties. Note that it is possible for $\text{contrib}(p,a)$ as defined above to decrease over time, e.g. if a publishes additional papers which bring $h(a) > cit(p)$. Next, we suggest a version of the Social h-index which is non-decreasing.

For each $i \leq h(a)$, consider the time at which a first achieved an h-index of i, and let $H^{(i)}(a) \subseteq P(a)$ be the set of papers with at least i citations at that time. For the *Progressive Social h-index* of author a, denoted $\text{Prog-Soc}^h(a)$, we define

$$\text{contrib}(p,a) = \sum_{\substack{i \leq h(a): \\ p \in H^{(i)}(a)}} \frac{i}{|H^{(i)}(a)|}.$$

Intuitively, the Progressive Social h-index assigns partial credit to a researcher's coauthors every time her h-index increases. We note that $\text{Prog-Soc}^h(a)$ is non-decreasing over time. That is, once a gets credit for contributing to another's success, that credit can not be overshadowed by future work. However, an exceptionally good paper could continue reaping rewards as a coauthor's h-index grows, if it remains one of the contributing papers.

Future work could analyze the AC game using the Instantaneous or the Progressive Social h-index as the players' utility functions, and compare the resulting behavior with that in the current work. Furthermore, while we focused on the context of academia, our approach can be applied to study collaborative behavior in other contexts such as business teams and collaborative design.

5.4.2 Limitations of Our Approach

In order to simplify analysis, we made several modeling assumptions that are not realistic. For example, we assumed that all citations for a paper are received immediately upon publication.

Taking a different citation model, such as a constant number of additional citations per paper per year as in [42], or a peak-decay model as suggested in [39], would complicate analysis but may lead to more realistic results. On the other hand, it may be that asymptotically these variations lead to the same behavior.

There is also an inherent limitation in modeling human relationships and interactions. The underlying premise that people can be modeled as rational agents is itself subject to debate. Even if we take that to be a reasonable model, there are many more factors at play in the real world of academia – e.g. geographic location, personal relationships, institutional loyalties, and academic competition – than can be captured by a simple mathematical model.

5.4.3 Significance and Impact

In this work, we have presented a game-theoretic approach to studying collaborative behavior in academia by modeling researchers as rational agents trying to maximize their academic success. Several publication models have been proposed in the bibliometric literature, but to our knowledge, ours is the first with the flexibility to model collaborative behaviors that may change over time in response to actions of and interactions with others. Our model makes it possible to simulate and therefore predict the growth of the academic community as a whole when individuals are driven by a specific objective.

The written policies of our academic institutions, as well as the unspoken rules and expectations of academia, inevitably shape the mentalities and goals of individual researchers, encouraging certain behaviors and discouraging others. An increased ability to understand the effects of these motivating forces will help policy-makers and academic leaders to make informed decisions that will stimulate the growth and progress of the academic community.

Chapter 6

Related Problems and Future Work

In this chapter, we explore three other problems related to the study of event-driven networks, and present some initial work.

6.1 Measuring Pairwise Influence

First, we build on the REWARDS model introduced in Chapter 3 to measure the influence that one node has on another based on the times of their respective activity.

Given two events ϕ and ψ generated by renewal processes Φ and Ψ, respectively, we say that the ordered pair (ϕ, ψ) are *consecutive* if $t(\phi) < t(\psi)$ and $\nexists\, t \in T_\Phi \cup T_\Psi$ such that $t(\phi) \leq t \leq t(\psi)$. We refer to the elapsed time between consecutive events ϕ and ψ as the (ϕ, ψ)*-gap*:

$$\mathrm{Gap}(\phi, \psi) = t(\psi) - t(\phi).$$

To identify influential relationships in a network, we could look for ordered pairs of nodes with many small gaps. However, this will bias the analysis towards nodes with higher activity rates, which are more likely to have small gaps. To compensate for this time scale bias, we follow a normalization procedure similar to that in Section 3.2.3.

Let $F^{\mathrm{Gap}}_{\Phi,\Psi}$ denote the limit distribution of gaps between consecutive event pairs for two independent renewal processes Φ and Ψ:[1]

$$F^{\mathrm{Gap}}_{\Phi,\Psi}(\tau) = \lim_{t\to\infty} Pr(\mathrm{Gap}(\phi, \psi) \leq \tau \mid (\phi, \psi) \text{ are consecutive}).$$

[1]The limit is well-defined except in certain cases when the support of the inter-arrival distributions for both Φ and Ψ have measure zero.

We define the *responsiveness* between two consecutive events ϕ and ψ to be

$$\mathrm{Resp}(\phi, \psi) = 1 - F^{\mathrm{Gap}}_{\Phi,\Psi}(\mathrm{Gap}(\phi, \psi)).^2$$

Note that Resp satisfies the uniformity property described in Section 3.2.2, i.e. that for independent renewal processes Φ and Ψ, randomly sampling the responsiveness across all consecutive event pairs will generate uniform random samples in $[0, 1]$. This normalization is scale-invariant (the responsiveness between any pair of consecutive events remains the same when time is stretched by a constant factor), which again makes our approach robust to differences in time scale between networks or between entities within the same network.

Next, we consider correlation of responsiveness among multiple pairs of consecutive events. Given a set Ω of consecutive event pairs, we define the collective responsiveness of Ω as $\mathrm{Resp}(\Omega) = 1 - p_{KS}$, where p_{KS} is the p-value from performing the Kolmogorov-Smirnov test on the individual responsiveness values, as described in Section 3.2.2. Larger values of $\mathrm{Resp}(\Omega)$ are a stronger indication that the corresponding processes are not independent.

Let $\mathcal{N} = (\mathcal{U}, \mathcal{E})$ be an event-driven network, and consider two nodes $u, u' \in U$. Then we define the responsiveness of u' to u as $\mathrm{Resp}(u, u') = \mathrm{Resp}(\Omega)$, where Ω is the set of all pairs of consecutive events between the renewal processes corresponding to the discrete-event sequences $\mathcal{E}_{(u,u')}$ and $\mathcal{E}_{u'}$, respectively. Intuitively, this measures whether there is a greater likelihood of activity from u' shortly after receiving information from u. In future work, we plan to explore the use of responsiveness to detect and measure influence in networks.

6.2 Innovation and Circulation

In Chapter 4 we suggested co-clustering as a way of identifying communities of individuals who participate in sharing the same content. However, that approach does not provide insight on the dynamics of information flow within communities or the roles of individual nodes. Various centrality measures have been suggested to indicate the relative importance of nodes in a network. These have traditionally been designed for static graphs, but recently several extensions have been proposed to accommodate graphs that change over time. In this work, we

[2] Similar to recency, we define responsiveness using $1 - CDF$ instead of CDF to match the linguistic intuition that higher responsiveness corresponds to shorter gaps.

apply our event-driven network model to better understand the roles that individuals play in the spread of information.

Consider an event-driven network $\mathcal{N} = (\mathcal{U}, \mathcal{E})$, where each event $\varepsilon \in \mathcal{E}$ has a single source node s_ε, and either is in response to an event received by s_ε or is an independently generated piece of new content, although which of the two scenarios applies may not be explicitly known. We would like to determine the most likely sources of new content, as well as measure the importance of each node in the diffusion process.

We formalize our problem with the following model: Let $\mathrm{rate}(u)$ denote the total rate of activity generated by node $u \in \mathcal{U}$. This activity can be decoupled into the rate of independently generated new content, denoted $\mathrm{innovation}(u)$, and the rate of activity which is in response to each incoming neighbor u', denoted $\mathrm{flow}(u', u)$. The goal is, given $\mathrm{rate}(u)$ for each node in the network, to infer the values of $\mathrm{innovation}(u)$ and $\mathrm{flow}(u, u')$ for each node and node pair, respectively. We additionally require as input an upper bound on the probability that u' responds to an event from u, which we denote by $p(u, u')$.

We frame the problem as a linear program:

Linear Program for Innovation and Flow

Input:

- for each node u, $\mathrm{rate}(u)$
- for each node pair (u, u'), $p(u, u')$

Variables:

- for each node u, $\mathrm{innovation}(u)$
- for each node pair (u, u'), $\mathrm{flow}(u, u')$

Constraints:

- $(\forall\, u \in \mathcal{U}) \quad \mathrm{innovation}(u) \geq 0$
- $(\forall\, u, u' \in \mathcal{U}) \quad \mathrm{flow}(u, u') \geq 0$
- $(\forall\, u, u' \in \mathcal{U}) \quad \mathrm{flow}(u, u') \leq p(u, u') \cdot \mathrm{rate}(u)$
- $(\forall\, u \in \mathcal{U}) \quad \mathrm{innovation}(u) + \sum_{u' \neq u} \mathrm{flow}(u', u) = \mathrm{rate}(u)$

Objective function:

- maximize $\sum_{(u,u')} \mathrm{flow}(u, u')$

The values of $\mathrm{innovation}(u)$ are uniquely determined, and are straight-forward to compute:

$$\mathrm{innovation}(u) = \max\left(\mathrm{rate}(u) - \sum_{u' \in \mathcal{U}} p(u', u) \cdot \mathrm{rate}(u'),\ 0\right).$$

The maximal value of the objective function represents the amount of network activity that

can be explained as responses to existing content. We refer to this as the *circulation* of the network, which is computed as follows:

$$\text{circulation}(\mathcal{N}) = \sum_{u \in \mathcal{U}} \text{rate}(u) - \text{innovation}(u).$$

However, the linear program may have many optimal solutions. Specifically, for each node u with $\text{rate}(u) < \sum_{u' \in \mathcal{U}} p(u', u) \cdot \text{rate}(u')$, the activity may be attributed to the incoming neighbors arbitrarily. We posit that this reflects a reality in information networks, that there is often redundancy in the information received by a node, in which case the source of the content to which a response should be attributed is inherently ambiguous. How, then, can we measure the influence a node has on the diffusion process?

We propose a measure which we term the *marginal circulation* of a node, which indicates how much more activity from the other nodes can be explained as responses than if the node were not there:

$$\delta_{circ}(u) = \text{circulation}(\mathcal{N}) - (\text{rate}(u) - \text{innovation}(u)) - \text{circulation}(\mathcal{N} - u).$$

It may also be of interest to compare the circulation between different networks. For this, we suggest the *circulation ratio*, which indicates the fraction of network activity that is responsive rather than innovative:

$$\xi_{circ}(\mathcal{N}) = \frac{\text{circulation}(\mathcal{N})}{\sum_{u \in \mathcal{U}} \text{rate}(u)}$$

We plan to explore this framework further in future work.

6.3 Cascade Partitioning

One term that has fallen into common use in the information diffusion literature is “cascade,” a sequence of connected node events induced through causal relationships. Several generative cascade models have been proposed [54, 36], as well as numerous studies analyzing the properties of known cascades [65, 61], but to the best of our knowledge, there has been no attempt to identify or extract cascades from unlabeled data. In this section, we use our event-driven network model to introduce several new problems relating to the study of cascades in information streams.

Let $\mathcal{N} = (\mathcal{U}, \mathcal{E})$ be an event-driven network. Given two events $\varepsilon, \varepsilon' \in \mathcal{E}$, we say that ε *precedes* ε' if $t_\varepsilon < t_{\varepsilon'}$ and $(\exists\, u \in \mathcal{U})$ such that $u \in R_\varepsilon$ and $u \in S_{\varepsilon'}$.

We define a *cascade* to be a set of events $\mathcal{C} \subseteq \mathcal{E}$ with a designated root event $\varepsilon^* \in \mathcal{C}$ such that for all $\varepsilon \in \mathcal{C},\ \varepsilon \neq \varepsilon^*$, there exists an event $\varepsilon' \in \mathcal{C}$ such that ε' precedes ε. A *cascade partition* of $\mathcal{N}$ is a set of cascades that partitions $\mathcal{E}$. The *cascade number* of a network $\mathcal{N}$ is the smallest non-negative integer k such that there exists a cascade partition of $\mathcal{N}$ of size k.

We say a cascade $\mathcal{C}$ is *simple* if for all $\varepsilon, \varepsilon' \in \mathcal{C}$, $s_\varepsilon \neq s_{\varepsilon'}$; that is, no source node appears more than once. A *simple cascade partition* of $\mathcal{N}$ is a set of simple cascades that partitions $\mathcal{E}$. The *simple cascade number* of a network $\mathcal{N}$ is the smallest non-negative integer k such that there exists a simple cascade partition of $\mathcal{N}$ of size k.

Theorem 6.1. *The cascade number of a network $\mathcal{N}$ can be computed in $O(|\mathcal{E}| \cdot |\mathcal{U}|)$ time.*

Theorem 6.1 is realized by Algorithm 6.1.

Algorithm 6.1 Greedy Algorithm for Cascade Partitioning

Input: An event-driven network $\mathcal{N} = (\mathcal{U}, \mathcal{E})$, with $\mathcal{E}$ in chronological order.

Output: A cascade partitioning $\mathcal{C}$ of M.

1: Initialize a collection of cascades $\mathcal{C} := \emptyset$ and an array of cascades A indexed by $\mathcal{U}$ with entries initialized to `null`.
2: For each event $\varepsilon \in \mathcal{E}$ (processed in chronological order):
   (a) If $A[s_\varepsilon] =$ `null` then create a new set $\mathcal{C} = \{\varepsilon\}$ and add $\mathcal{C}$ to $\mathcal{C}$; otherwise, let $\mathcal{C} = A[s_\varepsilon]$, add ε to $\mathcal{C}$, and then assign $A[s_\varepsilon] :=$ `null`.
   (b) For each node $r \in R_\varepsilon$, assign $A[r] := \mathcal{C}$.
3: Return $\mathcal{C}$, the set of cascades.

Theorem 6.2. *The decision problem for simple cascade partitioning is NP-hard.*

Theorem 6.2 can be proved via a reduction from Set Covering, a problem known to be NP-complete [52]. As an intermediate step, we construct a graph whose vertices are the events in $\mathcal{E}$ colored by their source nodes, and consider a problem of partitioning the graph into trees with no repeated colors. The full proof is omitted here.

Future work may consider cascade and simple cascade partitioning under different sets of constraints, or study the approximability of the simple cascade partitioning problem.

Chapter 7

Conclusions

7.1 Discussion of Themes

In this section, we review the central themes of our work, highlighting their relevance to each of the problems we have studied.

7.1.1 Graph Analysis

We have applied tools from Graph Theory to help address several different problems pertaining to event-driven networks. In Chapter 3, we construct an edge-weighted graph that represents the recency of communication between each pair of neighboring nodes in the network at a given point in time. We then propose the G-CORE algorithm, which uses a disjoint set data structure to search for subgraphs with a high concentration of recent activity. In Chapter 4, we efficiently search the set of possible co-clusterings of a matrix by traversing a pair of k-d trees, one for the rows and one for the columns, each co-clustering corresponding to a pair of maximal anti-chains on the trees. In Chapter 5, we find that the equilibria of the multi-player academic collaboration game include the set of all perfect matchings on the researchers. In Section 6.1, we measure the influence between a pair of nodes, which can be coupled with graph algorithms to study the structure of influence and hierarchy in networks. In Section 6.2, we study the effects that individual nodes have on the flow of information in a network by framing an optimization problem on a weighted graph. In Section 6.3, we determine that the simple cascade partitioning problem is NP-hard by a reduction from a partitioning problem on a vertex-colored graph.

7.1.2 Temporal Dynamics

Temporal dynamics are an important aspect of real-world networks. In Chapter 3, we propose a stochastic approach that models communication between nodes as renewal processes, and detect sets of nodes whose behavior is temporally correlated. In Chapter 4, we identify communities in social media as entities who participate in propagating many of the same memes over time. In Chapter 5, we model the world of academia as a repeated game, where every year the researchers may change their collaborations based on the results of theirs and other players' actions in previous years. This model can be used to simulate researchers' behavior over time and thus predict the growth of an academic field when researchers follow a particular set of strategies. In Section 6.1, we propose a new way to measure the influence of one node on another based on the likelihood that one's activity is in response to the other's. In Section 6.2, we suggest a model to study the generation and transfer of information across a network, which is inherently a temporal process. In Section 6.3, our definition of cascade enforces the temporal precedence of consecutive events.

7.1.3 Group Behavior

In real-world networks, individuals rarely operate in complete isolation. Many times, individual actions can be better understood in the context of group behavior. In Chapter 3, we look for correlations in the collective behavior of groups of nodes. In Chapter 4, we identify functional communities as groups of individuals with related behavior. In Chapter 5, we define a k-stable equilibrium as a strategy profile in which no group of k individuals would benefit from cooperatively choosing to deviate from their current strategies. We analyze the game for $k = 2$, and suggest exploring the case of $k > 2$ as future work. In Section 6.1, our measure of pairwise influence can be used to study the internal structure and dynamics of groups. In Section 6.2, we suggest the notion of circulation to capture how much of network activity is due to collective rather than individual behavior. In Section 6.3, cascades inherently entail the cooperative behavior of multiple nodes.

7.1.4 Attribution

In analysis of real-world networks, attribution can greatly improve the interpretability of results and lead to actionable information. In Chapter 3, we choose a statistical correlation test that identifies the exact nodes and events that are most responsible for the correlated behavior. In Chapter 4, as opposed to traditional clustering methods, our co-clustering approach attributes the similarity of elements in each row cluster to the sets of columns that many of them have in common. In Chapter 5, our game model attributes the citations that a paper receives to the collective effort invested by its authors. In Section 6.1, we attribute the influence of one node on another to the specific pairs of consecutive events that more likely have a causal relationship. In Section 6.2, we frame an optimization problem that determines how much of each node's activity should be attributed to responsive behavior, and suggest marginal circulation as a way to measure how much of the total flow in the network should be attributed to each individual node. In Section 6.3, we attribute each event that occurs in a network to the cascade which includes it.

7.1.5 Computational Realizability

Finally, we examine the computational issues that arise in applying our models and algorithms to real-world networks. In Chapter 3, we propose both a streaming local algorithm and a heuristic global algorithm that address the need for computational efficiency when dealing with large, high-volume communication networks. In Chapter 4, our CC-MACS algorithm leverages the sparsity of many real-world networks to run in time sub-linear in the size of the matrix. In Chapter 5, we discuss how assumptions about human rationality and the accessibility of information can affect the applicability of theoretic models to study real-world human behavior. In Section 6.1, by maintaining the distributions of inter-arrival times and responsiveness values for consecutive event pairs using dynamic distribution approximation methods, the collective responsiveness of one node to another can be computed efficiently in a streaming manner. In Section 6.2, we frame our problem as a linear program, whose solution can be computed efficiently. In Section 6.3, we discuss two variants of the cascade partitioning problem, presenting

an efficient algorithm for one and an NP-hardness proof for the other. Approximation algorithms would be needed to make the latter variant feasible in practice, which we suggest as a direction for future work.

7.2 Summary of Contributions

The first main contribution of our work is the formalization of a new framework for modeling event-driven networks. Our framework is flexible enough to model a wide variety of network types, including: direct pairwise communication such as email, phone, IP traffic, and face-to-face encounters; broadcast messages such as multi-recipient emails, blogs, and online social media; bipartite networks such as those that arise in recommender systems; and coauthorship and citation networks. It can easily model the addition of new nodes, and the formation and discontinuation of paths of information transfer. In the remainder of the dissertation, we apply this new framework to address a variety of problems that arise in the study of real-world networks.

In Chapter 3, we consider the task of detecting correlated events in communication networks. We first present the REWARDS (REneWal theory Approach for Real-time Data Streams) model, a new stochastic model for event-driven networks. Our approach aims to address the temporal variability present in communication networks, moving away from predominantly-used approaches that require an aggregation step or use a decay model with global parameters, which are sensitive to the time scale used for analysis. In particular, we give a formal definition of recency for renewal processes that is time scale-invariant, and propose a statistical test to identify the presence of recent correlated activity among a given set of entities. We then present algorithms to efficiently find such correlations in a network. The L-CORE algorithm detects correlations among outgoing activity from a single node, and is tailored for a distributed setting in which each node can perform the algorithm using only local information. All computations can be performed in a streaming manner with extremely low space requirements, making it ideal for nodes with a high-volume of traffic. The G-CORE algorithm simultaneously detects subsets of nodes exhibiting correlated activity in disparate parts of the network. A heuristic

version of the algorithm makes it computationally feasible for larger networks. Through experiments on synthetic and real-world data, we demonstrate that our approach effectively detects correlated events in communication networks.

In Chapter 4, we present a new approach to community discovery in information networks. As opposed to most existing work, which frames the problem as one of clustering well-connected vertices in a social network graph, we aim to identify functional communities as groups of individuals who participate in the dissemination of multiple common memes. Given a set of memes from an information network, we first construct a binary matrix, where the rows correspond to individuals in the network, the columns correspond to memes, and a 1-entry indicates that the individual participated in that meme. We then frame the problem as one of matrix co-clustering, simultaneously clustering the rows and columns of a matrix to reveal hidden structure. We propose a class of metrics that reward co-clusterings containing large, dense blocks, and then present the CC-MACS (Co-Clustering via Maximal Anti-Chain Search) algorithm, a new heuristic algorithm which efficiently searches the space of possible co-clusterings for one which maximizes the value of a given metric. The CC-MACS algorithm provides several benefits over previously proposed approaches: (1) the dense biclusters in the matrix need not have a block diagonal structure; (2) it explores the breadth of the search space rather than getting stuck at local optima; (3) the results are not dependent on user-specified parameters; and (4) it is designed to leverage the sparsity of many real-world networks, running in sub-linear time for sparse matrices. Finally, we apply the CC-MACS algorithm to discover functional communities in a large information network.

In Chapter 5, we aim to understand the mechanisms underlying academic collaboration using tools from the field of Game Theory. We begin by building a model for how researchers collaborate and how collaboration affects the number of citations a paper receives, supported by observations from a large real-world publication and citation dataset. Based on this model, we frame the world of academic research as a repeated game in which each researcher wants to maximize her h-index. We consider the single-player, two-player, and multi-player versions of the game, analyzing the asymptotic behavior and equilibria for each version. Our first main result is that for the two-player game, the researchers perform asymptotically better by collaborating, achieving linear growth of the h-index, than by publishing only independent work, for

which their h-index grows as the square root of the number of years they have been producing papers. Our second main result is that for the multi-player game, when players are constrained to following the same strategy every year, any strategy profile corresponding to a perfect matching on the set of researchers is an equilibrium; yet when strategies are allowed to change over time, the same strategy profiles are not equilibria. This highlights an important problem with the existing literature, most of which is based on models where collaboration strategies remain constant over time. Our game-theoretic approach provides a foundation for further study, which through analytical methods as well as simulation can help us to better understand the dynamics of collaborative systems.

7.3 The Big Picture

Our ability to gain actionable information from real-world network data is limited by the way the data is represented. The majority of known network analysis methods model networks as graphs, which opens the door to a suite of well-studied graph algorithms and theoretical tools. However, as computational technologies are being brought to bear against increasingly challenging real-world tasks, traditional models are no longer sufficient to capture the complexities and subtleties of the data. Some real-world networks can be modeled as time-evolving graphs; that is, graphs to which nodes and edges may be added or removed over time. This model makes sense when persistent relationships are explicitly observable, such as in an online social network. In many other real-world contexts, however, relationships can only be inferred through the observation of discrete events in continuous time, such as the sending of a message or the posting of online content. In such cases, flattening the data by constructing a sequence of snapshot or summary graphs at periodic intervals necessarily results in information loss, even before further computational methods have been applied. While some existing work has attempted to work directly with event-based data, the literature seems to lack a unifying model for event-driven networks upon which new analytical tools can be built. In Chapter 2, we laid the groundwork for such a model. In the subsequent chapters, we explored its application to address a variety of problems that arise in the study of real-world networks. It is our hope that this work will lead to improved methods and technologies to provide useful and actionable

information to human analysts in a wide variety of fields and contexts.

References

[1] I. B. Aban, M. M. Meerschaert, and A. K. Panorska. Parameter estimation for the truncated pareto distribution. *Journal of the American Statistical Assoc*, 101, 2006.

[2] A. Abbasi, J. Altmann, and J. Hwang. Evaluating scholars based on their academic collaboration activities. *Scientometrics*, 83(1):1–13, 2010.

[3] J. Abello, T. Eliassi-Rad, and N. Devanur. Detecting novel discrepancies in communication networks. In *10th IEEE International Conference on Data Mining*, pages 8–17, 2010.

[4] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient similarity search in sequence databases. In *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms*, FODO '93, pages 69–84, London, UK, UK, 1993. Springer-Verlag.

[5] L. Akoglu, M. McGlohon, and C. Faloutsos. Oddball: Spotting anomalies in weighted graphs. In M. Zaki, J. Yu, B. Ravindran, and V. Pudi, editors, *Adv. in Knowledge Disc. and Data Mining*, volume 6119 of *Lecture Notes in Comp. Sci.*, pages 410–421. Springer Berlin / Heidelberg, 2010.

[6] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 20–29, New York, NY, USA, 1996. ACM.

[7] L. A. N. Amaral, A. Scala, M. Barthélémy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences*, 97(21):11149–11152, 2000.

[8] Aristotle. *Metaphysics*. Clarendon Press, 1948. Reprint of the 350 BCE edition.

[9] B. Babcock, M. Datar, and R. Motwani. Sampling from a moving window over streaming data. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '02, pages 633–634, Philadelphia, PA, USA, 2002. Society for Industrial and Applied Mathematics.

[10] B. Babcock, M. Datar, R. Motwani, and L. O'Callaghan. Maintaining variance and k-medians over data stream windows. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '03, pages 234–243, New York, NY, USA, 2003. ACM.

[11] A.-L. Barabási. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039):207–211, May 2005.

[12] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[13] J. Candia, M. C. González, P. Wang, T. Schoenharl, G. Madey, and A.-L. Barabási. Uncovering individual and collective human dynamics from mobile phone records. *J. Physics A: Mathematical and Theoretical*, 41, June 2008.

[14] J. Cao, A. Chen, T. Bu, and A. Buvaneswari. Monitoring time-varying network streams using state-space models. In *INFOCOM 2009, IEEE*, pages 2721 –2725, 2009.

[15] A. Cardillo, G. Petri, V. Nicosia, R. Sinatra, J. Gómez-Gardeñes, and V. Latora. Evolutionary dynamics of time-resolved social interactions. arXiv:1302.0558v2, 2013.

[16] D. Chakrabarti, S. Papadimitriou, D. S. Modha, and C. Faloutsos. Fully automatic cross-associations. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 79–88, New York, NY, USA, 2004. ACM.

[17] D. Chakrabarti, Y. Zhan, and C. Faloutsos. R-mat: A recursive model for graph mining. In *SDM*, 2004.

[18] K.-P. Chan and A. W.-c. Fu. Efficient time series matching by wavelets. In *Proceedings of the 15th International Conference on Data Engineering*, ICDE '99, pages 126–133, Washington, DC, USA, 1999. IEEE Computer Society.

[19] Q. Chen, L. Chen, X. Lian, Y. Liu, and J. X. Yu. Indexable pla for efficient similarity search. In *Proceedings of the 33rd international conference on Very large data bases*, VLDB '07, pages 435–446. VLDB Endowment, 2007.

[20] M. D. Choudhury. How birds of a feather flock together on online social spaces. In *Grace Hopper Celebration of Women in Computing*, Atlanta, GA, USA, 2010.

[21] G. Cormode, M. Garofalakis, and D. Sacharidis. Fast approximate wavelet tracking on streams. In *Proceedings of the 10th International Conference on Extending Database Technology*, EDBT '06, pages 26–30, 2006.

[22] G. Cormode, Q. Ma, S. Muthukrishnan, and B. Thompson. Socializing the h-index. *Journal of Informetrics*, 7(3):718–721, 2013.

[23] D. J. de Solla Price. Networks of scientific papers. *Science*, 149(3683):510–515, 1965.

[24] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Frequency estimation of internet packet streams with limited space. In *Proceedings of the 10th Annual European Symposium on Algorithms*, ESA '02, pages 348–360, London, UK, UK, 2002. Springer-Verlag.

[25] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 89–98, New York, NY, USA, 2003. ACM.

[26] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proc. VLDB Endow.*, 1:1542–1552, August 2008.

[27] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 57–66, New York, NY, USA, 2001. ACM.

[28] N. Eagle, A. S. Pentland, and D. Lazer. Inferring friendship network structure by using mobile phone data. *Proc. of the Nat'l Academy of Sciences*, 106:15274–15278, Aug. 2009.

[29] L. Egghe. Theory and practise of the g-index. *Scientometrics*, 69(1):131–152, April 2006.

[30] K. Eren, M. Deveci, O. Küçüktunç, and Ü. V. Çatalyürek. A comparative analysis of biclustering algorithms for gene expression data. *Briefings in Bioinformatics*, 2012.

[31] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings of the 1994 ACM SIGMOD international conference on Management of data*, SIGMOD '94, pages 419–429, New York, NY, USA, 1994. ACM.

[32] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[33] P. Geurts. Pattern extraction for time series classification. In *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '01, pages 115–127, London, UK, UK, 2001. Springer-Verlag.

[34] Goh, K.-I. and Barabási, A.-L. Burstiness and memory in complex systems. *EPL*, 81(4):48002, 2008.

[35] C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, Aug. 1969.

[36] M. Granovetter. Threshold models of collective behavior. *The American Journal of Sociology*, 83(6):1420–1443, 1978.

[37] M. S. Granovetter. The strength of weak ties. *The American Journal of Sociology*, 78(6):1360–1380, 1973.

[38] S. Guha, N. Koudas, and K. Shim. Approximation and streaming algorithms for histogram construction problems. *ACM Trans. Database Syst.*, 31:396–438, March 2006.

[39] R. Guns and R. Rousseau. Simulating growth of the h-index. *Journal of the American Society for Information Science*, 60(2005):410–417, 2009.

[40] F. Heider. John Wiley & Sons, 1958.

[41] K. Henderson, T. Eliassi-Rad, C. Faloutsos, L. Akoglu, L. Li, K. Maruhashi, B. A. Prakash, and H. Tong. Metric forensics: a multi-level approach for mining volatile graphs. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 163–172, New York, NY, USA, 2010. ACM.

[42] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences of the United States of America*, 102(46):16569–16572, 2005.

[43] J. Holm, K. de Lichtenberg, and M. Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48:723–760, July 2001.

[44] D. Hume. *A Treatise of Human Nature*. Clarendon Press, 1896. Reprint of the 1739 edition.

[45] A. Ihler, J. Hutchins, and P. Smyth. Adaptive event detection with time-varying poisson processes. In *Proc. of 12th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, KDD '06, pages 207–216, 2006.

[46] A. Jain, J. M. Hellerstein, S. Ratnasamy, and D. Wetherall. A wakeup call for internet monitoring systems: The case for distributed triggers. *Proceedings of HotNets*, 2004.

[47] H. Jeong, Z. Néda, and A.-L. Barabási. Measuring preferential attachment in evolving networks. *Europhysics Letters*, 61(4):567–572, 2003.

[48] B. Jin. h-index: an evaluation indicator proposed by scientist. *Science Focus*, 1(1):8–9, 2006.

[49] P. W. Jones, A. Osipov, and V. Rokhlin. Randomized approximate nearest neighbors algorithm. *Proceedings of the National Academy of Sciences*, 108(38):15679–15686, 2011.

[50] G. A. A. Kahou, L. Grigori, and M. Sosonkina. A partitioning algorithm for block-diagonal matrices with overlap. *Parallel Comput.*, 34(6-8):332–344, July 2008.

[51] S. Kameshwaran, V. Pandit, S. Mehta, N. Viswanadham, and K. Dixit. Outcome aware ranking in interaction networks. In *Proceedings of ACM International Conference on Information and Knowledge Management*, pages 229–238, 2010.

[52] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, USA, 1972.

[53] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, SIGMOD '01, pages 151–162, New York, NY, USA, 2001. ACM.

[54] W. O. Kermack and A. G. McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society, London*, 115:700+, 1927.

[55] B. Klimt and Y. Yang. The enron corpus: A new dataset for email classification research. In J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, editors, *Machine Learning: ECML 2004*, volume 3201 of *Lecture Notes in Computer Science*, pages 217–226. Springer Berlin / Heidelberg, 2004.

[56] T. G. Kolda. Partitioning sparse rectangular matrices for parallel processing. In *Lecture Notes in Computer Science*, pages 68–79. Springer-Verlag, 1998.

[57] A. N. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *Giornale dell'Istituto Italiano degli Attuari*, 4:83–91, 1933.

[58] F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently supporting ad hoc queries in large datasets of time sequences. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, SIGMOD '97, pages 289–300, New York, NY, USA, 1997. ACM.

[59] A. Kumar, M. Sung, J. J. Xu, and J. Wang. Data streaming algorithms for efficient and accurate estimation of flow size distribution. In *Proceedings of the joint international conference on Measurement and modeling of computer systems*, SIGMETRICS '04/Performance '04, pages 177–188, New York, NY, USA, 2004. ACM.

[60] A. Lakhina, M. Crovella, and C. Diot. Characterization of network-wide anomalies in traffic flows. In *Proc. of the 4th ACM SIGCOMM Conf. on Internet Measurement*, IMC '04, pages 201–206, 2004.

[61] K. Lerman and R. Ghosh. Information contagion: An empirical study of the spread of news on digg and twitter social networks. In *Proceedings of 4th International Conference on Weblogs and Social Media (ICWSM)*, 2010.

[62] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 497–506, New York, NY, USA, 2009. ACM.

[63] J. Leskovec, D. Chakrabarti, J. Kleinberg, and C. Faloutsos. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In *Proceedings of the 9th European conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD'05, pages 133–145, Berlin, Heidelberg, 2005. Springer-Verlag.

[64] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *In KDD*, pages 177–187. ACM Press, 2005.

[65] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs: Patterns and a model. In *Proceedings of the 7th SIAM International Conference on Data Mining (SDM)*, 2007.

[66] L. Li, B. A. Prakash, and C. Faloutsos. Parsimonious linear fingerprinting for time series. *Proc. VLDB Endow.*, 3:385–396, September 2010.

[67] X. Lian, L. Chen, J. X. Yu, J. Han, and J. Ma. Multiscale representations for fast pattern matching in stream time series. *IEEE Trans. on Knowl. and Data Eng.*, 21:568–581, April 2009.

[68] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[69] L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A*, 390(6):1150–1170, 2011.

[70] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, Dec. 2007.

[71] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 1(1):24–45, Jan. 2004.

[72] S. Muthukrishnan, V. Poosala, and T. Suel. On rectangular partitionings in two dimensions: Algorithms, complexity, and applications. In *Proceedings of the 7th International Conference on Database Theory*, ICDT '99, pages 236–256, London, UK, UK, 1999. Springer-Verlag.

[73] S. Navathe, S. Ceri, G. Wiederhold, and J. Dou. Vertical partitioning algorithms for database design. *ACM Trans. Database Syst.*, 9(4):680–710, Dec. 1984.

[74] M. Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46:323–351, Sept.-Oct. 2005.

[75] C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *Proc. of the 9th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, KDD '03, pages 631–636, New York, NY, USA, 2003. ACM.

[76] T. Palpanas, V. Kalogeraki, and D. Gunopulos. Online distribution estimation for streaming data: Framework and applications. In *SEBD*, pages 430–438, 2007.

[77] R. Pang, M. Allman, V. Paxson, and J. Lee. The devil and packet trace anonymization. *SIGCOMM Comput. Commun. Rev.*, 36:29–38, January 2006.

[78] M. Pidd. *Computer simulation in management science*. John Wiley & Sons, Inc., New York, NY, USA, 1984.

[79] C. E. Priebe, J. M. Conroy, D. J. Marchette, and Y. Park. Scan statistics on enron graphs. *Computational and Mathematical Organization Theory*, 11:229–247, Oct. 2005.

[80] A. Rapoport. Spread of information through a population with socio-structural bias i: Assumption of transitivity. *Bulletin of Mathematical Biophysics*, 15(4):523–533, 1953.

[81] K. V. Ravi Kanth, D. Agrawal, and A. Singh. Dimensionality reduction for similarity searching in dynamic databases. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, SIGMOD '98, pages 166–176, New York, NY, USA, 1998. ACM.

[82] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Commun. ACM*, 43(12):45–48, Dec. 2000.

[83] E. M. Rogers. *Diffusion of Innovations*. Free Press, New York, 1st edition, 1962.

[84] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *Annals of Mathematical Statistics*, 27(3):832837, 1956.

[85] A. Rubinstein. Equilibrium in supergames with the overtaking criterion. *Journal of Economic Theory*, 21(1):1–9, 1979.

[86] A. Rubinstein. Comments on the interpretation of repeated games theory. In J. J. Laffont, editor, *Advances in Economic Theory*, volume 1, pages 175–181. Cambridge University Press, 1992.

[87] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43 – 49, feb 1978.

[88] A. Sandryhaila and J. M. F. Moura. Eigendecomposition of block tridiagonal matrices. *ArXiv e-prints*, June 2013.

[89] U. Sharan and J. Neville. Temporal-relational classifiers for prediction in evolving domains. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 540–549, Washington, DC, USA, 2008. IEEE Computer Society.

[90] R. Sulo, T. Berger-Wolf, and R. Grossman. Meaningful selection of temporal resolution for dynamic networks. In *Proc. of 8th Workshop on Mining and Learning with Graphs*, MLG '10, pages 127–136, New York, NY, USA, 2010. ACM.

[91] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 687–696, 2007.

[92] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *5th IEEE Int'l Conf. on Data Mining*, pages 418–425, 2005.

[93] A. Tanay, R. Sharan, and R. Shamir. Biclustering algorithms: A survey. In *Handbook of Computational Molecular Biology*.

[94] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22:215–225, April 1975.

[95] H. Tong, Y. Sakurai, T. Eliassi-Rad, and C. Faloutsos. Fast mining of complex time-stamped events. In *Proceedings of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 759–768, New York, NY, USA, 2008. ACM.

[96] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[97] J. Wu, S. Lozano, and D. Helbing. Empirical study of the growth dynamics in real career h-index sequences. *Journal of Informetrics*, 5(4):489–497, 2011.

[98] Y. Xie, V. Sekar, D. A. Maltz, M. K. Reiter, and H. Zhang. Worm origin identification using random moonwalks. In *Proc. of the 2005 IEEE Symposium on Security and Privacy*, pages 242–256, 2005.

[99] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, SFCS '82, pages 160–164, Washington, DC, USA, 1982. IEEE Computer Society.