COMMUNITY DETECTION AND ITS APPLICATIONS IN UNDERSTANDING DYNAMICS OF SOCIAL NETWORKS

Xiaoyan Lu

Submitted in Partial Fullfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

Approved by:
Dr. Boleslaw K. Szymanski, Chair
Dr. Gyorgy Korniss
Dr. Jianxi Gao
Dr. Malik Magdon-Ismail

Department of Computer Science
Rensselaer Polytechnic Institute
Troy, New York

[May 2019]
Submitted March 2019

© Copyright 2019
by
Xiaoyan Lu
All Rights Reserved

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

The research in this thesis was supported in part by the Army Research Office grant W911NF-12-1-0546, by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 (the ARL Network Science CTA), and by the Office of Naval Research Grant No. N00014-15-1-2640. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies either expressed or implied of the Army Research Laboratory or the US Government.

First and foremost, I would like to thank my advisor, Prof. Boleslaw K. Szymanski, for his guidance throughout my tenure as a graduate student. It was a great fortune to work with him during my doctoral study. This work would not have been possible without his guidance and his commitment to excellence in research. I learned from him not only the fundamentals of conducting scientific research but also a hardworking and passionate attitude on my work and life.

I would like to thank my collaborators, Prof. Eyuphan Bulut, Prof. Chris Gaiteri and Prof. Jianxi Gao for their insightful guidance. Prof. Gao introduced me to the computational physics methods for complex network analysis. Prof. Bulut introduced me to work on scale-free P2P network construction. Prof. Chris Gaiteri worked with me on the collaboration recommendation system for biomedical researchers based on the co-authorship and molecule interaction networks. I appreciate the opportunity of working with Dr. Konstantin Kuzmin, Dr. Mingming Chen, and Robert Paluch during my doctoral study. I am grateful for the friendship and support of my fellow collaborators.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Malik Magdon-Ismail, Prof. Gyorgy Korniss, and Prof. Jianxi Gao for kindly serving on my doctoral committee, and for their helpfulness and willingness to answer questions that arose during my time as a graduate student.

Finally, I would like to extend my love and gratitude to my family: Hui, Qiong and Xilei. I would not be where I am today without their love and support.

ABSTRACT

Emergent social phenomena like radicalization, civil unrest, and opinion migration can be explained by the accumulative influences between individuals in a social network. Such emergent phenomena are often described as "nobody saw them coming" because of their explosive dynamics, and yet they have a profound impact on our society. Since these dynamics are significantly affected by the social network topology, there is a strong desire to study the interplay between the emergent dynamics and the underlying social network structure. In this doctoral thesis, we focus on the community structure in social networks and study its role in the prediction of emergent network dynamics.

Community structures are observed across a wide variety of networks, including the World Wide Web, Internet, research collaboration, transportation, social and biochemical networks. Community detection aims at discovering the partition of the network nodes into groups such that the edges inside each group are generally more numerous than the edges across them. Modularity is perhaps the most widely used quality metric to evaluate the partition of network nodes. Despite being one of the most widely used state-of-the-art community detection approaches, modularity maximization suffers from the resolution limit problem which arises due to the implicit dependence of the modularity definition on a constant (explicitly defined in the generalized modularity as a resolution parameter). In this thesis, we uncover importance of this dependence using random graph theories to explain the resolution limit of modularity. Specifically, we establish the asymptotic theoretical upper and lower bounds on the resolution parameter of generalized modularity for the modularity maximization to recover community structure correctly, which is the first work connecting the resolution limit of modularity with the random graph models. The unified theory shows that, in generalized modularity maximization, the well-formed communities whose densities are smaller than the resolution parameter are split into multiple clusters; but the well-formed communities whose background inter-community edge densities are larger than the resolution parameter are merged into one large component. This result reveals a "plateaus" problem that no resolution parameter that avoids such damaging splits and mergers exists when the density of any well-formed community is smaller than the background inter-community edge density among some other well-formed communities. Therefore, we propose a progres-

sive agglomerative heuristic algorithm based on a statistical hypothesis testing framework that systematically increases the resolution parameter to partition a graph recursively. The statistical hypothesis testing checks if the partition found by each branch of recursion is significant. If it is, this recursion branch continues splitting the current graph at the next level; otherwise, this recursion branch terminates, accepting the null hypothesis that the current subgraph is already a community.

While most community detection algorithms take unweighted graph as input by default, they can be extended to accept the edge weights. In this thesis, we also show that the appropriately assigned edge weights can improve the quality of the detected communities. We propose an edge weighting scheme to avoid the bias of modularity maximization towards merging well-formed small communities into large ones. Our proposed edge weighting scheme works in a semi-supervised fashion: a regression model penalized by the merging of small ground truth communities is trained to convert the local edge features into edge weights. Experimental results show that, in addition to modularity maximization, the five different state-of-the-art approaches are also significantly improved by the edge weights produced by our model.

Besides the maximization of modularity and its generalized version, an alternative approach to detect communities is the statistical inference to fit the generative graph model to the observed network data. The degree-corrected stochastic block model is one such random graph model, generating different network partitions, ranging from traditional assortative communities to disassortative structures. It does not impose any constraints on the mixing pattern of the resulting block assignments, thus the return of the traditional assortative community structures is not guaranteed. On top of the degree-corrected stochastic block model, we propose a generative random graph model which puts a constraint on nodes' internal degree ratio. This model stabilizes the inference of block model, avoiding inference algorithms like Markov chain Monte Carlo to get trapped in the local optima of the log-likelihood. Unlike the modularity maximization algorithm which always attempts to find traditional assortative communities, in this regularized model, one single regularization parameter controls the mixing patterns discovered from the given network.

In the context of information propagation, the community structures play an essential role in facilitating the local spread of information because the community members are more likely to accept inputs from each other than from the outsiders. On the other hand, the

community structures slow down global information diffusion due to trapping the messages in dense regions and thus preventing global penetration. For this reason, we formulate a virality prediction framework for global online news using the community structure of online news media. The goal is to classify the viral news at their early stage of spreading. Our virality prediction framework considers every news a cascade in the network formed by online media sites. When an online media site reports a piece of news, it gets infected by the corresponding cascade. Based on the statistical survival analysis, our model defines the probability of infection along the network edges given the propagation delay. Since the news generally spreads among the media sites of similar cultural backgrounds, our model assumes each media site is affiliated with some communities, and the news cascade reaches each community with a certain probability. The mathematical model, after simplification, becomes a graph representation learning model which finds the node embeddings from the news cascades. Given the node embeddings of the early adopters, the F1 score of the viral news prediction produced by our approach outperforms the results of feature-based approaches by almost 20% on the Global Database of Events, Language, and Tone (GDELT) dataset. We parallelize the corresponding graph representation learning algorithm for both shared memory and distributed memory machines. The parallelized stochastic gradient descent algorithm is shown to scale well to 10K+ cores in the IBM Blue Gene/Q supercomputer at RPI (ranked as one of the fastest world-wide IBM Blue Gene configuration in academia). Moreover, in order to facilitate the information spread in peer-to-peer (P2P) networks, we proposed a P2P overlay network construction algorithm which maintains the power-law distribution of nodes degree while peers dynamically join and leave the P2P network.

Finally, we study the patterns of the polarization evolution by analyzing millions of roll-call votes in the legislative branches of the United States. Community structure of the members in these legislative branches is a critical factor for the development of political polarization. We proposed a social dynamical model explaining the formation of polarization observed in real legislative voting data and derived the tipping points of the dynamical system which provides early warning signals when the system is close to the bifurcation. Our dynamical model successfully explains the directions of polarization change in 28 out of the past 30 U.S. Congresses. The hidden variable in the dynamical model, called the polarization utility, is shown to correlate well with critical historical events such as the civil rights movement and Super PACs.

CHAPTER 1
INTRODUCTION AND BACKGROUND

1.1 Introduction

The arrival of the information age has profoundly changed the way how people socialize. With the rise of a wide variety of online communities, the rapid spread of information is able to reach a large number of audiences and lead to emergent public responses to various social, economic and political events in a short period. How does the information sharing in social networks lead to the emergent social phenomena? Can we use scientific models to explain and predict the viral trends which are often missed by a human observer? The availability of a wealth of data can help us answer these questions and understand how information dissemination fuels emergent social phenomena.

Accumulative influences between individuals are often used to explain the adoption of information in online networks. A user is likely to accept a message from her friends and re-share it to the others who share a tie with her. Such successive re-sharing of the same message in a snowball manner results in viral information cascades. The theories of mathematical epidemiology have been established to model the spread of contagious diseases in the early 1900s [1]. The widely used SIS/SIR epidemic models [2], [3] and its' variants are shown to be successful predictive tools to model the spread of disease in the population. Although these models are widely used to explain the disease spreading processes, there exist some barriers to model the information propagation in a social network using the SIS/SIR models. First, it is unclear when the recovery from infection occurs. After an individual has accepted the information, there is no specific sign when the user explicitly discards or forgets this information. Also, social networks are shown to be extremely sparse [4], composed of many small groups [5]. In order to accurately model the information propagation, the underlying network topology should be taken into consideration as well.

In this thesis, we discuss the interplay between the information cascades and the underlying network topology. Specifically, we focus on the impact of community structures on information propagation. Community structures are widely observed in social, biological and technological networks. In the context of information propagation, the community structures play an essential role in facilitating the local spread of information because the

community members are more likely to accept inputs from each other than from the outsiders. On the other hand, the community structures slow down global information diffusion due to trapping the messages in dense regions and thus preventing global penetration.

Community detection aims at discovering the division of the network nodes into clusters such that the edges inside each cluster are generally more numerous than the edges across them. Despite being one of the most widely used state-of-the-art community detection approaches, modularity maximization suffers from the resolution limit problem which arises due to the implicit dependence of the modularity definition on a constant (explicitly defined in the generalized modularity as a resolution parameter). The modularity maximization tends to merge the small, well-formed communities into one large component while splitting one large well-formed community into smaller clusters inappropriately. Some variants of the modularity function have been proposed to either resolve this problem [6], [7] or to enable detection of communities at different scales. A popular choice for the latter is the generalized modularity of Reichardt and Bornholdt [8], which scales the discovered community sizes according to a simple resolution parameter.

In this thesis, we explain the resolution limit problem [9] by defining the asymptotic theoretical upper and lower bounds on the resolution parameter of generalized modularity [10] for the modularity maximization algorithm to recover community structure correctly. To the best of our knowledge, this is the first work connecting the resolution limit of modularity with the random graph models. The unified theory shows that, in generalized modularity maximization, the well-formed communities whose densities are smaller than the resolution parameter are split into multiple clusters; but the well-formed communities whose background inter-community edge densities are larger than the resolution parameter are merged into one large component. This result reveals a "plateaus" problem that no resolution parameter that avoids such damaging splits and mergers exists when the density of any well-formed community is smaller than the background inter-community edge density among some other well-formed communities. Therefore, based on a statistical hypothesis testing framework, we propose a progressive agglomerative heuristic algorithm that systematically increases the resolution parameter to partition a graph recursively. The statistical hypothesis testing checks if the partition found by some branch at each recursion level is significant. If it is, this recursive branch continues splitting the current graph at the next level; otherwise, this recursion branch terminates, accepting the null hypothesis that the

current subgraph is already a well-formed community.

We investigate the performance of several state-of-the-art community detection algorithms on real and synthetic networks. While these community detection algorithms often take unweighted graph as input, they can be extended to accept the edge weights for community detection. A novel edge weighting scheme [7] is proposed to improve the quality of communities discovered by the state-of-the-art algorithms. The edge weighting scheme [7] avoids the bias of modularity maximization towards merging well-formed small communities into a large one. Our proposed edge weighting scheme works in a semi-supervised fashion: a regression model penalized by the merging of small ground truth communities is trained to convert the local edge features into edge weights. The experimental results show that, in addition to modularity maximization, the five different state-of-the-art approaches are also significantly improved by the assigned edge weights, even if their optimization goals differ from maximizing the modularity.

Besides the maximization of modularity and its generalized version, an alternative approach to detect communities is the statistical inference to fit the generative graph model such as the stochastic block model to the observed network data. The standard stochastic block model is a generative model of the graph in which nodes are organized as blocks and edges are placed between nodes independently at random, with a probability determined by the block assignments of the endpoints. Hence, the nodes in the same block are statistically indistinguishable from each other. The degree-corrected stochastic block model extends the standard stochastic block model by incorporating the node degrees, which improves the performance of community detection. Yet, it does not impose any constraints on the mixing pattern of the resulting block assignments, thus the return of the traditional assortative community structures is not guaranteed. On top of the degree-corrected stochastic block model, we propose a generative random graph model which puts a constraint on nodes' internal degree ratio in the objective function to stabilize the inference of block model in sparse networks [11]. Thus, our model prevents inference algorithms like Markov chain Monte Carlo from getting trapped in the local optima of the log-likelihood. Unlike the modularity maximization algorithm which always attempts to find traditional assortative communities, the inference of this regularized stochastic block model controls the mixing patterns discovered in the given network. Therefore, the inference algorithm like Markov chain Monte Carlo can reliably find the desired community structures in different optimization trials.

The diversity of the communities in which the early adopters reside is a good predictor of cascade virality [12]. For this reason, we formulate information cascades model [13] based on the community structures of the online news media. In order to predict the viral news at the early stage of spreading, each news is considered a cascade in the network formed by news media sites. Once the media site reports a piece of news, the corresponding media site nodes in the network get infected by the cascade of this news. Our model defines the probability of infection along the network edges given the propagation delay using the statistical survival analysis. It incorporates the community-level signals which reduce the complexity of the model and improves the virality prediction accuracy. The inference algorithms of our model have been successfully parallelized for both shared memory [13] and distributed memory machines [14]. On the Global Database of Events, Language, and Tone (GDELT) dataset, the F1 score of the viral news prediction produced by our approach outperforms the results of feature-based approaches by almost 20%. Furthermore, in order to facilitate the information spread in peer-to-peer (P2P) networks, we proposed a distributed P2P overlay network construction algorithm [15] which maintains the power-law distribution of nodes degree while peers dynamically join and leave the P2P network.

Community structure plays a critical role in the development of social polarization. While social influence models [16]–[20] study different aspects of social polarization, they share some common assumptions about human social behavior [21], including the following: (i) individuals iteratively update their views to reach consensus with their neighbors in a social network; (ii) the tolerance of conflicting views is limited in social context, so frequent active disagreements usually break social ties. These assumptions indicate that the loyalty to one's group usually leads to the conformity with views of the group's majority [22], and such conformity tightens social ties within the group. Therefore, community structure is critical for understanding the evolution of social polarization.

We study the patterns of the political polarization evolution by analyzing millions of roll-call votes in the legislative branches of the United States [23]. The agent-based social dynamical model is proposed to explain the formation of polarization observed in real legislative voting data. The derived the tipping points of the dynamical system provides an early warning signal of political polarization when the system is close to the bifurcation. Our dynamical model successfully explains the directions of polarization change in 28 out of the past 30 elected U.S. Congresses. The hidden variable in the dynamical model, called

the polarization utility, correlates well with critical historical events such as the civil rights movement and Super PACs.

1.2 Contributions

This thesis centers around theory and practice of community structures detection at scale and its applications in understanding the dynamics of social networks. The specific novel contributions of this thesis include:

1. An edge weighting scheme [7] that avoids the resolution limit of the modularity maximization, improving the quality of communities discovered by the state-of-that-art community detection algorithms.

2. Asymptotic theoretical lower and upper bounds on the resolution parameter of generalized modularity for the modularity maximization algorithm to recover community structure correctly [10] using the random graph properties, which connecting the resolution limit of modularity with the random graph models.

3. An agglomerative heuristic algorithm which recursively divides the network into subgraphs to detect communities at different scales and a statistical hypothesis testing framework ensuring the significance of the partitions at each recursion level [10]

4. A regularized stochastic block model which controls the mixing pattern of the resulting community structures [11] that avoids getting the inference algorithms trapped in the local optima of the log-likelihood.

5. A distributed algorithm for dynamic limited scale-free network construction that at each instance of its evolution adheres to the desired Power Law of node degrees [15] so that the network topology optimizes information propagation speed while being robust against failures.

6. A model of information cascades [13] based on survival analysis and community detection to improve the prediction of the viral information cascades.

7. A parallelized stochastic gradient descent algorithm for graph representation learning [14] which scales well on both shared memory and distributed memory machines.

8. A dynamical model [23] quantifying the evolution of polarization in the U.S. Congresses elected in the past six decades. It successfully predicts the direction of polarization changes in 28 out of 30 elected U.S. Congresses. The hidden model parameter, polarization utility, correlates well with significant political or legislative changes happening at the same time.

CHAPTER 2
COMMUNITY DETECTION IN COMPLEX NETWORKS

2.1 Introduction

The study of complex networks reveals the universal principles underlying the network topology. One of the most significant features of complex networks is the community structure, which is widely observed in the social, biological and technological networks. Detecting such community structures can be viewed as partitioning of the network into sub-modules in which the nodes are more densely connected to each other than to the nodes in the rest of the network. Many important tasks, such as data extraction, link prediction, network evolution analysis, and graph mining are based on the community structures discovered in these networks. Therefore, community detection finds its application in a wide variety of research fields, ranging from neurobiology to statistical physics.

Modularity [24] is perhaps the most widely used quality metric for network partitions. It measures the difference between the observed fraction of edges within a community and this fraction expected in a random graph with the same number of nodes and the same degree sequence. Due to the implicit dependence of modularity definition on a constant (explicitly defined in the generalized modularity as a resolution parameter), however, modularity maximization suffers from the resolution limit problem [9], [25] which causes the small, well-formed communities to be merged into a large community since this merging increases modularity. In Section 2.2, we review the definition of modularity and its generalized version and weighted version, followed by the discussion of the resolution limit of modularity maximization.

Besides modularity maximization approaches, an alternative method to detect communities is to use the statistical inference to fit a generative random graph model to the observed

Portions of this chapter previously appeared as:

X. Lu, K. Kuzmin, M. Chen, and B. K. Szymanski, "Adaptive modularity maximization via edge weighting scheme," *Inf. Sci.*, vol. 424, pp. 5568, 2018.

X. Lu and B. K. Szymanski, "Asymptotic resolution bounds of generalized modularity and statistically significant community detection," 2019, *arXiv:1902.04243*

Portions of this chapter are submitted as:

X. Lu and B. K. Szymanski, "Regularized Stochastic Block Model for robust community detection in complex networks," submitted for publication.

network data. The maximization of generalized modularity is shown to be equivalent to the maximum likelihood estimation of the degree-corrected planted partition model on the same graph when the resolution parameter takes a particular value [26]. Such equivalence shows that modularity maximization is a special case of the statistical inference of random graph models. Modularity maximization assumes all the communities in the network are similar which is the major cause for the resolution limit problem as we will see in Section 2.3. Section 2.2.2 reviews the definitions of the stochastic block model, its degree-corrected variant and the planted partition model.

Based on the random graph properties, we establish the asymptotic upper and lower bounds for the resolution parameter of the generalized modularity for the modularity maximization algorithm to recover community structure correctly [10]. To the best of our knowledge, this is the first result connecting the well-known resolution limit of modularity with the random graph theories. Given a resolution parameter within the established range, we show that the communities with different densities can still be detected by maximizing the generalized modularity, regardless whether the equivalence between generalized modularity maximization and the maximum likelihood estimation of the stochastic block model holds or not. When the resolution parameter of the generalized modularity is larger than the upper bound we developed, some well-formed communities are likely to be spread among multiple clusters to increase the generalized modularity; but when the resolution parameter of generalized modularity is smaller than the lower bound we developed, some communities are inappropriately merged into one large cluster. In Section 2.3, we present these theoretical results in details.

Our findings shed light how to adapt the generalized modularity to networks in which there is no universal resolution parameter to detect all communities. Resolution limits of the generalized modularity arise when there are the non-overlapping lower and upper bounds of the resolution parameter in the different subgraphs of the networks. Therefore, either some large well-formed communities are split into multiple clusters or some well-formed communities are inappropriately merged into one large cluster. The issue is analogous to identifying mountains that are located at different plateaus; using a single altitude either would miss the lower mountains, or would treat the higher peaks as one mountain.

To address the problem mentioned above, in Section 2.4, we propose a progressive agglomerative heuristic algorithm that systematically increases the resolution parameter.

The algorithm recursively splits the clusters found by the previous branch of recursion to detect smaller communities. As the recursion proceeds, the algorithm gradually increases the resolution parameter for high-resolution community detection in local subgraphs of the network. The algorithm proceeds until the found partition is not statistically significant. Compared with the algorithm using a uniform resolution parameter, our approach avoids getting trapped by the resolution limit and does not require multiple re-estimation of the resolution parameter [26] which can be computationally prohibitively costly for large networks.

As we discussed above, given an input network, modularity maximization is one special case of the statistical inference of the degree-corrected stochastic block model. The degree-corrected stochastic block model is able to generate a wide range of structures, ranging from the traditional assortative community structure to the disassortative structures such as the core-periphery and bi-partite networks. The model itself does not impose any constraint on which structure is returned. Instead, the inference algorithm returns the most likely partition of the network according to the maximum likelihood estimations. Thus, the inference algorithms, such as Markov chain Monte Carlo, are likely to get trapped by the local optima of the log-likelihood, corresponding to either assortative or disassortative structures. We propose a regularized model which uses one single model parameter to control the resulting structure assortativity by inference. This regularized model is presented in Section 2.5.

In Section 2.6, we propose an efficient edge weight scheme to improve the quality of the communities discovered by the state-of-art community detection algorithms [7]. The edge weighting scheme samples a small set of connected ground truth communities from a synthetic network sharing similar properties as the input networks. The changes of modularity upon merging these sampled communities are used as the penalty terms of a regression model which converts the local edge features to the corresponding edge weights. Hence, instead of assigning the edge weights in an ad-hoc style, our proposed edge weighting scheme searches for the most appropriate edge weights automatically. Although the proposed edge weight scheme is based on the modularity maximization only, the edges weights produced by our model also significantly improve many other state-of-the-art community detection algorithms. Section 2.6 presents the edge weighting scheme and the experimental results on real and synthetic networks.

2.2 Related work

2.2.1 Modularity and resolution limit

Modularity maximization is one of the state-of-the-art methods for community detection that has gained popularity in the last decade. The goal of the modularity maximization is to discover community structure in a network by maximizing the modularity, defined as

$$Q = \frac{1}{2m} \sum_{r} \sum_{\{i,j\} \in r} \left(A_{ij} - \frac{k_i k_j}{2m} \right), \tag{2.1}$$

where A_{ij} is the number of edges between nodes i and j and $\{i, j\} \in r$ denotes all the pairs of nodes i and j in the same block r, k_i is the degree of node i and m is the total number of edges in the network. In an unweighted undirected network, $2m = \sum_i k_i$ because every edge has exact two endpoints by definition.

Modularity can be also written in an alternative way as

$$Q = \sum_{r} \left[\frac{m_r}{m} - \left(\frac{\kappa_r}{2m} \right)^2 \right], \tag{2.2}$$

where m_r is the number of edges with both endpoints inside the community r, $\kappa_r = \sum_{i \in r} k_i$ is the sum of the degrees of nodes in community r, and m is the total number of edges in the network.

The generalized modularity of Reichardt and Bornholdt [8] incorporates a simple resolution parameter γ and is defined as a function of γ as follows:

$$Q(\gamma) = \frac{1}{2m} \sum_{r} \sum_{\{i,j\} \in r} \left(A_{ij} - \gamma \frac{k_i k_j}{2m} \right). \tag{2.3}$$

The definition of the generalized modularity does not provide the value of the resolution parameter. The alternative of the generalized modularity is

$$Q(\gamma) = \sum_{r} \left[\frac{m_r}{m} - \gamma \left(\frac{\kappa_r}{2m} \right)^2 \right], \tag{2.4}$$

The modularity can be naturally extended to the networks with weighted edges by replacing the count of edges with the sum of their weights. Hence, the weighted modularity

is defined as

$$Q(\gamma) = \frac{1}{2W} \sum_{r} \sum_{\{i,j\} \in r} \left(W_{ij} - \gamma \frac{w_i w_j}{2W} \right), \tag{2.5}$$

$$Q = \frac{1}{2W} \sum_{r} \sum_{\{i,j\} \in r} \left(W_{ij} - \frac{w_i w_j}{2W} \right), \tag{2.6}$$

where W is the sum of weights of edges in the entire graph, $W_{i,j}$ is the weight of the edge between nodes i and j, w_i is the sum of the weights of edges adjacent to node i. The original definition of modularity is a special case of the weighted version when the weight of every edge is 1.

For the modularity maximization to be able to find a community r with m_r edges inside, the following inequality must hold [9],

$$m_r \geq \sqrt{\frac{m}{2}}. \tag{2.7}$$

where m_r is the number of the edges inside community r, and m is the total number of edges of the entire graph. In large networks with millions of edges, the number of edges in most communities is often smaller than this lower bound. In such cases, the small, well-formed communities are often merged into a large cluster to increase the modularity. This problem is also referred to as the resolution limit of modularity.

2.2.2 Stochastic Block Model

The standard stochastic block model is a generative model of the graph in which nodes are organized as blocks and edges are placed between nodes independently at random, with a probability determined by the block assignments of the endpoints. Hence, the nodes in the same block are statistically indistinguishable from each other by definition. The standard stochastic block model assumes the number of edges between nodes i and j is independently distributed Bernoulli with mean ω_{g_i,g_j}, where for any node l, g_l denotes the block assignment of node l. The stochastic block model is fully specified by the block assignment g_i for each node i and the edge probability ω_{rs} for all pairs of blocks (r, s). Since the standard stochastic block model considers nodes in the same block statistically indistinguishable, the most likely block assignment often groups the nodes of similar degrees in a block, resulting in lower and higher-degree blocks, rather than the traditional community

structures. Reference [27] extends the standard stochastic block model by incorporating the node degrees. This simple yet effective modification improves the performance of the models for statistical inference of group structure in the real-world networks, by allowing nodes in a community to have broad range of degrees. To simplify technical derivations, authors in [27] replace the Bernoulli distribution by the Poisson one. Compared to many community detection algorithms proposed in an ad-hoc manner, the statistical inference from stochastic block model is a more systematic approach which discovers modular structures with theoretical guarantees [28]; thus, it has gained significant attention in recent years.

Formally, let $\mathbf{A}$ be the adjacency matrix whose component A_{ij} denotes the number of edges between nodes i and j in an unweighted undirected multigraph. In a network with self-loop edges (edges connecting a node to itself), a node i with k self-loop edges is represented by the diagonal adjacency matrix element $A_{ii} = 2k$. Multi-edges and self-loop edges are practical in certain networks such as the web network where a web page may contain multiple hyperlinks pointing to other pages and to itself. Such edges are less common in social networks. However, most social networks are very sparse, so the impact of multi-graphs and self-loop edges can be neglected.

Suppose the number of edges between two different nodes i and j follows the Poisson distribution with mean η_{ij} and the number of self-loop edges at node l follows the Poisson distribution with mean $\frac{1}{2}\eta_{ll}$. The likelihood of generating $\mathbf{A}$ is

$$\log P(\mathbf{A}|\{\eta_{ij}\}) = \prod_i \frac{(\frac{1}{2}\eta_{ii})^{A_{ii}/2}}{(\frac{1}{2}A_{ii})!} e^{-\eta_{ii}/2} \prod_{i<j} \frac{\eta_{ij}^{A_{ij}}}{A_{ij}!} e^{-\eta_{ij}}. \tag{2.8}$$

Note that η_{ij} is defined as the number of edges here and not, as customary, the probability because multi-edges are allowed in our model.

In an unweighted undirect network, after ignoring all terms independent of the η_{ij}s, the log-likelihood above simplifies to

$$\log P(\mathbf{A}|\{\eta_{ij}\}) = \frac{1}{2}\sum_{ij}\Big(A_{ij}\log\eta_{ij} - \eta_{ij}\Big). \tag{2.9}$$

The degree-corrected stochastic block model [27] assumes $\eta_{ij} = \omega_{g_i,g_j}\theta_i\theta_j$ where for any node l, θ_l is a parameter associated with this node and g_l is its block assignment. Given a partition of the network, i.e. its block assignments $\{g_i\}$, the posterior maximum likelihood

estimates of θ_i and ω_{rs} are

$$\hat{\theta}_i = \frac{k_i}{\kappa_{g_i}}, \qquad \hat{\omega}_{rs} = m_{rs}, \tag{2.10}$$

where k_i is the degree of node i, κ_r is the sum of the degrees of all nodes in a block r, and m_{rs} is the total number of edges between blocks r and s, or twice the number of edges in r if $r = s$. Formally,

$$\kappa_r = \sum_{i \in r} k_i, \qquad m_{rs} = \sum_{i \in r} \sum_{j \in s} A_{ij}, \tag{2.11}$$

where $i \in r$ denotes all nodes in the blocks r. For simplicity, we define m_c as the number of edges with both endpoints in community c, so $m_{cc} = 2m_c$, and m is defined as the total number of edges in the entire network.

Given the MLEs of θ_k and $\omega_{g_i g_j}$, the expected number of edges between nodes i and j can be written as

$$\eta_{ij} = m_{g_i g_j} \frac{k_i}{\kappa_{g_i}} \frac{k_j}{\kappa_{g_j}}. \tag{2.12}$$

Therefore, maximizing the log-likelihood in Eq. 2.9 is equivalent to maximizing

$$\log P(\mathbf{A}|\{\eta_{ij}\}) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{\kappa_r \kappa_s}. \tag{2.13}$$

This criterion turns to have an information-theoretic interpretation when it is presented in the alternative form

$$\log P(\mathbf{A}|\{\eta_{ij}\}) = \sum_{rs} \frac{m_{rs}}{2m} \log \frac{\frac{m_{rs}}{2m}}{\frac{\kappa_r}{2m}\frac{\kappa_s}{2m}}, \tag{2.14}$$

which is equal to the Kullback-Leibler divergence [29] between the probability of observing an edge (i, j) pointing from block r toward s, i.e. $P[i \in r, j \in s] = m_{rs}/2m$, and the corresponding probability in a random graph with the same degree sequence, i.e. $P'[i \in r, j \in s] = (\kappa_r/2m)(\kappa_s/2m)$.

The degree-corrected planted partition model [26] is a special case of the degree-corrected stochastic block model in which the expected number of edges η_{ij} in a graph has the form

$$\eta_{ij} = \begin{cases} \omega_1 \theta_i \theta_j & \text{if } g_i = g_j, \\ \omega_0 \theta_i \theta_j & \text{otherwise.} \end{cases} \tag{2.15}$$

Hence, the degree-corrected stochastic block model reduces to the degree-corrected planted

partition model when the parameters $\omega_{rr} = \omega_1$ for all r and $\omega_{rs} = \omega_0$ for all pairs r, s such that $r \neq s$.

Newman [26] shows the maximization of the generalized modularity is equivalent to the maximum-likelihood estimation (MLE) of the degree-corrected planted partition model [27] on the same graph. In other words, the partition of the network which best fits the degree-corrected planted partition model to the network data also maximize the generalized modularity given the corresponding resolution parameter.

The degree-corrected stochastic block model [26], [27] incorporates the heterogeneous node degrees in the real networks. Reference [26] assumes the number of edges between nodes i and j follows the Poisson distribution with mean $\eta_{ij} = \omega_{g_i,g_j}\frac{k_i k_j}{2m}$ where m is the total number of edges in the network. The denominator $2m$ is only optional, but it is convenient for technical derivation as shown below. This definition is actually equivalent to the definition in Eq. 2.12 but the parameter ω_{rs} here has a different meaning. To distinguish this definition from the original definition of expected number of edges in [27], in the rest of this paper, we call ω_{rs} the *density* of the edges between blocks r, s. When $r = s$, ω_{rs} corresponds to the intra-community edge *density*; otherwise, ω_{rs} is called the inter-community edge *density*. Accordingly, the (r, s)-th element of the *density* matrix Ω is ω_{rs}. Besides, we assume that a group of nodes in the same block of the stochastic block model represents a single community. Following this line of reasoning, the degree-corrected planted partition model [26] defines the expected number of edges between two nodes, i and j, as

$$\eta_{ij} = \begin{cases} \omega_1 \frac{k_i k_j}{2m} & \text{if } g_i = g_j, \\ \omega_0 \frac{k_i k_j}{2m} & \text{otherwise.} \end{cases} \tag{2.16}$$

Hence, the *density* matrix Ω of this degree-corrected planted partition model only involves two possible values, i.e., $\Omega = \{\omega_1, \omega_0\}$.

Substituting Eq. 2.16 into Eq. 2.9, the log-likelihood for the degree-corrected planted partition model, after a small amount of manipulation, becomes

$$\log P(\mathbf{A}|\Omega, \mathbf{g}) = B\left[\frac{1}{2m}\sum_r \sum_{\{i,j\}\in r}\left(A_{ij} - \gamma\frac{k_i k_j}{2m}\right)\right] + C + D, \tag{2.17}$$

where $\{i, j\} \in r$ denotes all the pairs of nodes i and j in the same block r, B and C

are constants that depend on $\Omega = \{w_1, w_0\}$ but not on the block assignment $\mathbf{g} = \{g_i\}$, specifically,

$$\begin{aligned} B &= m\left(\log \omega_1 - \log \omega_0\right) \\ C &= m\left(\log \omega_0 - \omega_0\right) \\ D &= \sum_i k_i \log k_i - \log\left(2m\right) \end{aligned} \tag{2.18}$$

and the resolution parameter γ in Eq. 2.17 is a resolution parameter defined as

$$\gamma = \frac{\omega_1 - \omega_0}{\log \omega_1 - \log \omega_0}. \tag{2.19}$$

Note that the log-likelihood here is exact for undirected unweighted multigraphs; it has a slightly different form than the derivation in [26] where constant term D is ignored as it is independent from the parameters Ω and $\mathbf{g}$. Since B, C and D are constants given the value of Ω, the maximum likelihood estimation of $\mathbf{g}$ is then equivalent to maximizing the generalized modularity

$$Q(\gamma) = \frac{1}{2m} \sum_r \sum_{\{i,j\} \in r} \left(A_{ij} - \gamma \frac{k_i k_j}{2m} \right). \tag{2.20}$$

Since maximizing the log-likelihood in Eq. 2.17 is equivalent to maximizing the generalized modularity with resolution parameter defined in Eq. 2.19, it shows an exact equivalence between maximization of the generalized modularity that includes a specific resolution parameter and the degree-corrected planted partition model.

It is worth noting that the resolution parameter γ must satisfy Eq. 2.19 to guarantee the equivalence between the generalized modularity maximization and the maximum likelihood estimation of degree-corrected planted partition model. This is a drawback because the correct value of resolution parameter in Eq. 2.19 is unknown before we can estimate ω_1 and ω_0. Given a network generated by the degree-corrected planted partition model, however, one needs the ground truth communities to calculate ω_1 and ω_0. That being said, one must make an initial guess of the γ value so that the generalized modularity maximization algorithm can detect communities with it. But this initial guess of γ may not satisfy Eq. 2.19.

Reference [26] proposes an iterative algorithm to find the optimal value of γ which

maximizes the log-likelihood of Eq. 2.17. It makes an initial guess of the γ value to detect communities by maximizing the generalized modularity. Given these detected communities, the γ value gets updated by Eq. 2.19 using the maximum likelihood estimates of ω_1 and ω_0 which are equal to

$$\hat{\omega}_1 = \frac{2\sum_r m_r}{\sum_r \kappa_r^2/2m}, \qquad \hat{\omega}_0 = \frac{2m - 2\sum_r m_r}{2m - \sum_r \kappa_r^2/2m}, \tag{2.21}$$

where m_r is the number of edges with both endpoints in the community r, i.e. $m_r = \frac{1}{2}m_{r,r}$ as defined by Eq. 2.11 and κ_r is the sum of degrees of the nodes in the community r. The iterative algorithm repeats the following two steps until the γ value becomes stable:

- Given the current γ, find the block assignment which maximizes the generalized modularity $Q(\gamma)$, i.e., $\mathbf{g}^{\text{new}} = \arg\max_{\mathbf{g}} Q(\gamma)$.
- Given the current block assignment $\mathbf{g}$, calculate the maximum likelihood estimates of ω_1 and ω_0 using Eq. 2.21 and update the γ value by Eq. 2.19 accordingly.

2.2.3 Metrics comparing network partitions

The computation of modularity does not require the ground truth labels of the detected communities in a network. When the ground truth communities are available, the following metric are also generally used to evaluate the quality of the detected communities.

Let the ground truth partition of the graph be denoted as $GN = \{g_1, g_2, \ldots\}$ where g_i is the set of nodes in a single ground truth community, and the detected communities be denoted as $C = \{c_1, c_2, \ldots\}$ where c_i is the set of nodes in single community discovered the community detection algorithm. The following evaluation metrics measure the similarity between the produced partition C and ground truth partition GN.

Variation of Information (VI) [30] measures the similarity between C and GN based on information theory

$$VI(C, GN) = H(C) + H(GN) - 2I(C, GN) \tag{2.22}$$

where $I(C, GN) = H(C) + H(GN) - H(C, GN)$ is the *Mutual Information*, and $H()$ is the

entropy function defined as

$$H(C) = -\sum_{c_i \in C} p(c_i) \log p(c_i) = -\sum_{c_i \in C} \frac{|c_i|}{|V|} \log \frac{|c_i|}{|V|} \tag{2.23}$$

$$\begin{aligned} H(C, GN) &= -\sum_{c_i \in C, g_i \in GN} p(c_i, g_i) \log p(c_i, g_i) \\ &= -\sum_{c_i \in C, g_i \in GN} \frac{|c_i \cap g_i|}{|V|} \log \frac{|c_i \cap g_i|}{|V|} \end{aligned} \tag{2.24}$$

Normalized Mutual Information (NMI) [31] is defined as

$$NMI(C, GN) = \frac{2I(C, GN)}{H(C) + H(GN)} \tag{2.25}$$

Adjusted Mutual Information (AMI) [32] is defined as

$$AMI(U, Q) = \frac{I(C, GN) - E\left[I(C, GN)\right]}{max\left\{H(C), H(GN)\right\} - E\left[I(C, GN)\right]} \tag{2.26}$$

where $E\left[I(C, GN)\right]$ is the expected mutual information between two random clusterings as defined in [32].

F-measure [31] is given as

$$\textit{F-measure}(C, GN) = \frac{1}{|V|} \sum_{c_i \in C} |c_i| \max_{g_i \in GN} \frac{2|c_i \cap g_i|}{|c_i| + |g_i|} \tag{2.27}$$

Adjusted Rand Index (ARI) [33] computes the similarity by comparing all pairs of nodes that are assigned to the same or different communities in C and GN. Specifically, ARI is defined as

$$ARI(C, GN) = \frac{\sum_{ij} \binom{|c_i \cap g_j|}{2} - \frac{\left[\sum_i \binom{|c_i|}{2} \sum_j \binom{|g_j|}{2}\right]}{\binom{|V|}{2}}}{\frac{1}{2}\left[\sum_i \binom{|c_i|}{2} + \sum_j \binom{|g_j|}{2}\right] - \frac{\left[\sum_i \binom{|c_i|}{2} \sum_j \binom{|g_j|}{2}\right]}{\binom{|V|}{2}}}. \tag{2.28}$$

Modularity Density [6] is a measure of the quality of communities in a network.

Like the original modularity, it does not need the ground truth. The formal definition is

$$Q_{ds} = \sum_{c_i \in C} \left[\frac{m_{c_i}}{m} d_{c_i} - \left(\frac{\kappa_{c_i}}{2m} d_{c_i} \right)^2 - \sum_{\substack{c_j \in C \\ c_j \neq c_i}} \frac{m_{c_i,c_j}}{2m} d_{c_i,c_j} \right] \tag{2.29}$$

$$d_{c_i} = \frac{2m_{c_i}}{n_{c_i}(n_{c_i}-1)} \qquad d_{c_i,c_j} = \frac{m_{c_i,c_j}}{n_{c_i}(n_{c_i}-1)} \tag{2.30}$$

where n_{c_i} is the number of nodes in community c_i, d_{c_i} is the internal density of community c_i, and d_{c_i,c_j} is the pair-wise density between community c_i and community c_j.

2.3 Asymptotic resolution bounds of generalized modularity

Merging two communities, r and s, results in the following equalities: the total number of edges inside the merged community $m_{r \cup s} = m_r + m_s + m_{r,s}$ and the sum of degrees of the nodes inside the merged community $\kappa_{r \cup s} = \kappa_r + \kappa_s$. Hence, given the formalization of the generalized modularity in Eq. 2.4, the optimization algorithm is able to detect two well-formed communities r and s if the change of generalized modularity from merging r and s is negative, leading to the inequality

$$\Delta Q(\gamma) = \frac{m_{rs}}{m} - \gamma \frac{\kappa_r \kappa_s}{2m^2} < 0, \tag{2.31}$$

which can be rewritten in the alternative way as

$$\kappa_r \kappa_s > 2 \frac{m_{rs} m}{\gamma}. \tag{2.32}$$

Otherwise, when the $\Delta Q(\gamma) \geq 0$, communities r and s are merged to increase $Q(\gamma)$. Clearly, one can always increase γ so that Eq. 2.32 holds for any small κ_r and κ_s. But, a large γ may result in inappropriate split of some communities. To see this point, consider a community t comprised of two sets of nodes t' and $t'' = t - t'$ with sums of degrees κ' and κ'' respectively, and m'' edges between nodes in t' and t''. To avoid splitting community t into t' and t'', the inequality

$$\kappa' \kappa'' < 2 \frac{m'' m}{\gamma} \tag{2.33}$$

must hold. Given Eq. 2.32 and Eq. 2.33, we have

$$\frac{\kappa'\kappa''}{m''} < \frac{2m}{\gamma} < \frac{\kappa_r\kappa_s}{m_{rs}}. \tag{2.34}$$

A simple and straightforward explanation of the resolution limit found by Fortunato et al. [11] is that in realistic networks the above inequality may not hold. Indeed, it is likely that in a large network there exist communities r and s with small κ_r and κ_s and community t with large sum of node degrees $(\kappa' + \kappa'')$, such that the inequality above does not hold because $\frac{\kappa_r\kappa_s}{m_{rs}} < \frac{\kappa'\kappa''}{m''}$, giving rise to resolution limit anomaly.

Newman [26] recently has shown that the maximization of generalized modularity on planted partition networks is optimal, because it is equivalent to the maximum-likelihood estimation of the degree-corrected planted partition model as long as the correct resolution parameter is chosen.

Here we consider the extended planted partition model which allows the *density* matrix Ω to have different values on the diagonal - each community has its own *density* ω_r as the r-th element on diagonal of Ω, while keeping the background inter-community *density* ω_0, the same as defined in degree-corrected planted partition model in Eq. 2.16. The definition of the expected number of edges between nodes i and j in this case becomes

$$\eta_{ij} = \begin{cases} \omega_r \frac{k_i k_j}{2m} & \text{if } g_i = g_j = r, \\ \omega_0 \frac{k_i k_j}{2m} & \text{if } g_i \neq g_j. \end{cases} \tag{2.35}$$

Given the nodes' degrees k_i and k_j, $\frac{k_i k_j}{2m}$ is the expected number of edges between nodes i and j in the graph ensembles generated by the configuration model [34]. Thus, the *density* ω_r can be treated as the relative ratio of the expected number of edges in the extended planted partition model to the corresponding values in configuration model. Given the definition in Eq. 2.35, the corresponding *density* matrix of T communities is

$$\Omega = \mathrm{diag}_T\left(\{\omega_r\}\right) + \left(J_T - I_T\right)\omega_0, \tag{2.36}$$

where $\mathrm{diag}_T(\{\omega_r\})$ represents the diagonal matrix of size $T \times T$ with the intra-community edge *densities* $w_1, w_2, \ldots, w_T$ on the diagonal, J_T and I_T are correspondingly a matrix with all elements equal to 1 and the identity matrix of size $T \times T$.

The log-likelihood of this extended planted partition model is

$$\log P(\mathbf{A}|\Omega, \mathbf{g}) = \frac{1}{2} \sum_{\{i,j\} \in r} \beta_r \Big(A_{ij} - \gamma_r \frac{k_i k_j}{2m} \Big) + C + D, \tag{2.37}$$

where the expressions of C and D here are the same as mentioned above in Eq. 2.18, but the γ parameter is replaced by a set of community-specific parameters γ_r and the β_r defined as

$$\beta_r = \log \omega_r - \log \omega_0, \qquad \gamma_r = \frac{\omega_r - \omega_0}{\log \omega_r - \log \omega_0}. \tag{2.38}$$

When all communities has the same edge density ω_r, the log-likelihood of the extended model has the same form as the degree-corrected planted partition model defined by Eq. 2.17.

The maximum likelihood estimates of the intra-community and inter-community edge *densities* are

$$\hat{\omega}_r = \frac{2m_r}{\kappa_r^2/2m}, \qquad \hat{\omega}_0 = \frac{2m - \sum_r 2m_r}{2m - \sum_r \kappa_r^2/2m}, \tag{2.39}$$

where m_r is the number of the edges with both endpoints in the community r, κ_r is the sum of degrees of the nodes in community r.

For the extended planted partition model with degree correction, the number of edges between different communities r and s can be approximated by its expectation

$$m_{rs} \approx \sum_{i \in r, j \in s} \omega_0 \frac{k_i k_j}{2m} = \omega_0 \frac{\kappa_r \kappa_s}{2m}, \tag{2.40}$$

and for a community t with *density* parameter ω_t, the number of edges between its two subsets of nodes t' and t'' is

$$m'' \approx \sum_{i \in t', j \in t''} \omega_t \frac{k_i k_j}{2m} = \omega_t \frac{\kappa' \kappa''}{2m}. \tag{2.41}$$

Using the approximations above, the inequality in Eq. 2.34 holds if

$$\forall t : \frac{2m}{\omega_t} < \frac{2m}{\gamma} < \frac{2m}{\omega_0}. \tag{2.42}$$

Suppose there are T communities with the *density* parameters monotonically ordered by

their indices, the value of the resolution parameter γ should satisfy

$$\omega_0 < \gamma < \omega_1 < \omega_2 < .. < \omega_T, \tag{2.43}$$

so that all T communities can be detected. This result connects the resolution limit of generalized modularity with the random graph models.

Eq. 2.43 indicates that the suitable γ value should be as small as possible to avoid splitting of any well-defined community. However, it should not be smaller than the background inter-community *density* ω_0, otherwise, when γ is larger than the *density* parameter of some loose community t, this community t is likely to be split. For graphs that are likely generated by the extended planted partition model and for which the resolution bounds of Eq. 2.43) are satisified, the generalized modularity is still a good quality measure for community detection, especially when $\omega_r \gg \omega_0$ for every community r, as it leaves a sufficient interval for selecting γ which will prevent the resolution limit from arising.

Motivated by the above observation, we evaluate the performance of generalized modularity in a network generated by the extended planted partition model. The network comprises of ten communities, each has ten nodes and a *density* ω_r illustrated by the red vertical lines. The background inter-community *density* ω_0 is chosen as 0.17, a value much smaller than any community *densities*. The performance of community detection is measured by the normalized mutual information (NMI) metric [31], that compares the detected partition $\mathbf{c}$ with the ground truth $\mathbf{q}$ using the Eq. 2.25.

As Fig. 2.1(b) shows, the generalized modularity performs well with resolution parameter in the interval $\gamma \in [1.8, 4.2]$, generally matching the derived theoretical bound $\omega_0 < \gamma < \omega_1$. As γ approaches either side of the bound, the resolution parameter is either higher or lower than desired. Since the asymptotic bounds here are derived by approximating the number of edges by the corresponding expectation in the random graph model (Eq. 2.40, (2.41), as γ is getting closer and closer to either ω_0 or ω_1, the asymptotic results are getting further and further away from the true values, causing the NMI score to drop. This scenario is illustrated in Fig. 2.1(a) where every community r is placed at the height corresponding to its *density* ω_r. When the resolution parameter is between the asymptotic bounds, the communities above can be successfully detected. However, when the resolution parameter is larger than the *density* of some communities, those communities are at risk to

Table 2.1. The maximum-likelihood estimates of w_0 and w_1 and the interval of the resolution parameter that detects communities with NMI score larger than 90% in a range of empirical networks of n nodes and m edges. The number of communities q used for each network is the ground truth value generally accepted in the previous literature. The optimal γ were published in [26].

Network	n	m	q	γ	$(\hat{w}_0, \hat{w}_1)$	90% interval
Karate club	34	78	2	0.78	(0.26, 1.74)	(0.63, 1.37)
Dolphin social	62	159	2	0.59	(0.12, 1.42)	(0.58, 2.0)
Les Miserables	77	254	6	1.36	(0.35, 2.83)	(1.15, 1.54)
College football	115	614	11	2.27	(0.36, 5.11)	(1.78,5.61)

be split into smaller parts.

The asymptotic bounds also agree with the empirical observation made without any theoretical justification by the authors of [35]–[37], that the most suitable values of the resolution parameter γ occurs in the most stable plateaus in experiments. In addition, our results explain why [26] found that the statistical inference of the resolution parameter converges quickly to the desired value in small networks. It is because, once the resolution parameter falls into the range of $\omega_0 < \gamma < \omega_1$, if feasible, the community detection results become stable and the inference algorithm immediately yields the final resolution parameter after this stage.

For a range of empirical networks of n nodes and m edges, including the Karate club network [38], the dolphin social network [39], the network of interactions between fictional characters in the novel Les Miserables [40] and the network of games between American college football teams in the year 2000 [40], we compute the maximum-likelihood estimates of the background edge density w_0 and the lowest intra-community edge density w_1, fitting an extended planted partition model given the number of communities q and the optimal value of γ obtained by the statistical inference of [26]. We compute the modularity maximization results with a total of 100 different resolution parameters in the range $[0.2, 3w_1/2]$ and compare these results with the communities produced with γ from this range. The subrange which produces an NMI score higher than 90% is shown in Table 2.1. As shown in Fig. 2.2, although these empirical networks are not generated by the extended planted partition model, the stable intervals of resolution parameter lie inside the asymptotic lower and upper bounds.

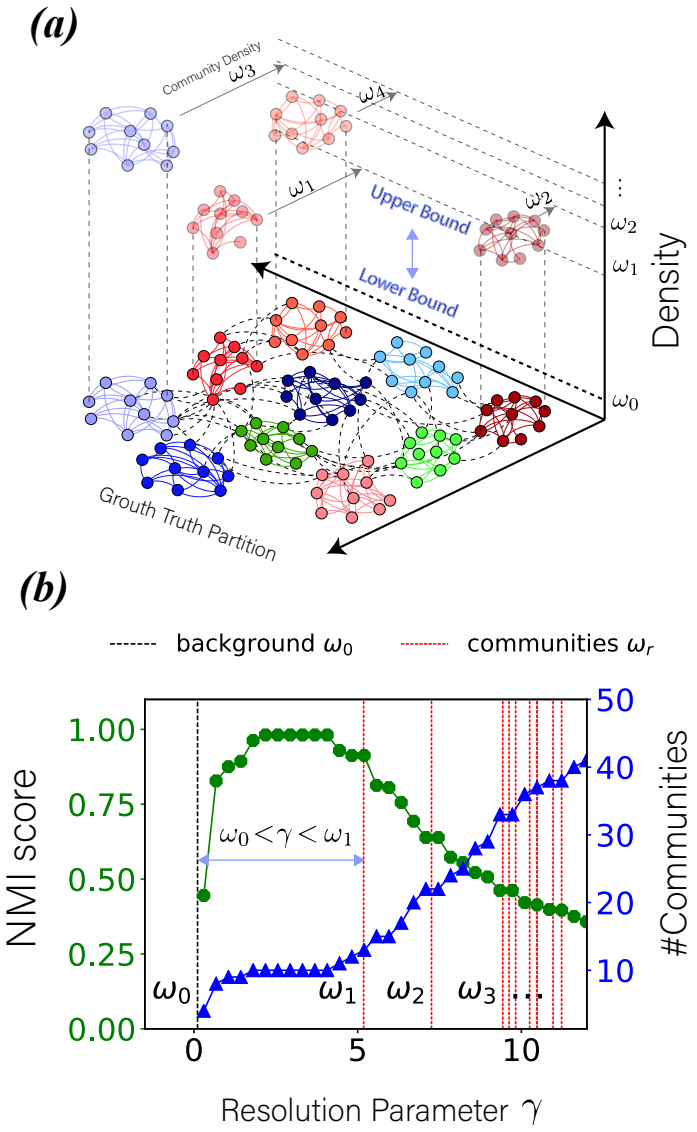


Fig. 2.1. The generalized modularity performs well with resolution parameter in the interval $\gamma \in [1.8, 4.2]$, matching the derived theoretical bound $\omega_0 < \gamma < \omega_1$. When γ approaches either side of the bound, the resolution scale is either higher or lower than desired. (a) Network structure and community *density* represented by the heights, the node color represents ground truth communities and inter-communities edges are in black dashed lines; (b) the NMI scores and the number of detected communities in relation to the resolution parameter.

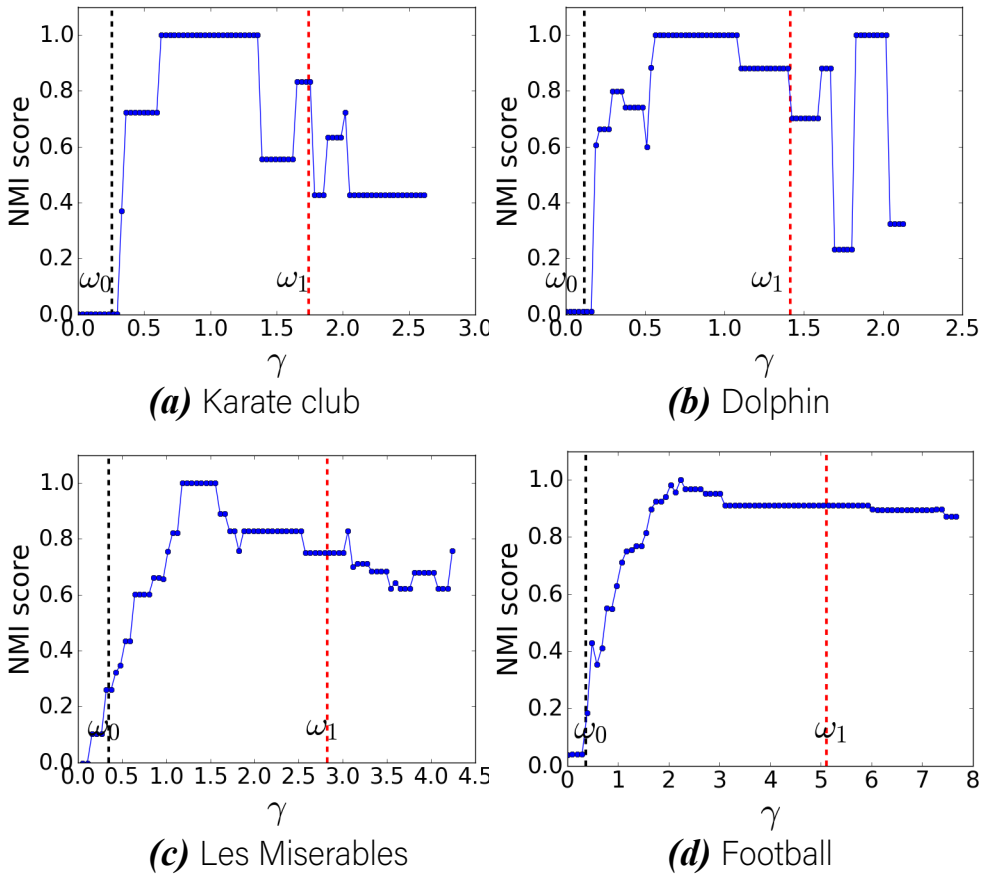


Fig. 2.2. The alignment between the communities detected by generalized modularity maximization and the optimal γ values listed in [26]. Although the empirical networks are not generated by the extend planted partition model, maximizing the generalized modularity is optimal when the resolution parameter takes values that lie in the interval $[\omega_0, \omega_1]$. This phenomenon is also captured purely experimentally and without any theoretical justification in [35]–[37].

2.4 Statistically significant community detection

The planted partition model and its extension introduced here are all special cases of the stochastic block model. The derived asymptotic resolution bounds can be extended to the networks generated by the degree-corrected stochastic block model. Following the notations in [26], the number of edges between nodes i and j in the degree-corrected stochastic block model follows the Poisson distribution with the mean defined as

$$\eta_{ij} = \omega_{g_i g_j} \frac{k_i k_j}{2m}, \tag{2.44}$$

where for node l, g_l is the block assignment of this node, k_l is its degree and m is the total number of edges in the network.

The number of edges between two communities r and s in this case is approximated by

$$m_{rs} \approx \sum_{i \in r, j \in s} \omega_{rs} \frac{k_i k_j}{2m} = \omega_{rs} \frac{\kappa_r \kappa_s}{2m}, \tag{2.45}$$

where ω_{rs} denotes the (r, s)-th element of the *density* matrix. The number of edges between two subsets of nodes t' and $t'' = t - t'$ inside a community t is approximated by

$$m_{t't''} \approx \sum_{i \in t', j \in t''} \omega_{tt} \frac{k_i k_j}{2m} = \omega_{tt} \frac{\kappa_{t'} \kappa_{t''}}{2m}, \tag{2.46}$$

where ω_{tt} is the t-th diagonal element in the *density* matrix.

Using the same resolution inequality of Eq. 2.34, these approximations lead to the range

$$\max_{r \neq s} \omega_{rs} < \gamma < \min_{r} \omega_{rr} \tag{2.47}$$

within which a uniform γ value avoids the resolution limit trap.

In a network generated by the degree-corrected stochastic block model with $\omega_{rs} > \omega_{tt}$ for some $r \neq s$ and t, Eq. 2.47 indicates that a uniform resolution parameter is not sufficient for the recovery of communities r, s and t.

The classical example of resolution limit trap is presented in [25] where an undirected unweighted network contains three communities: two cliques and one ER random graph, and every two communities are connected by one single edge. Suppose each clique includes 6 nodes and the ER random graph contains 100 nodes and 956 edges. Given the three communities, the posterior estimation of the *density* matrix Ω of a stochastic block model, i.e., $\hat{\omega}_{rs} = \frac{2m_{rs}m}{\kappa_r \kappa_s}$ for each communities r and s, is

$$\Omega = \begin{bmatrix} 1.03 & 0.03 & 0.03 \\ 0.03 & 57.94 & 1.93 \\ 0.03 & 1.93 & 57.94 \end{bmatrix}, \tag{2.48}$$

where the first row and column corresponds to the random graph while the remaining rows and columns correspond to the two cliques respectively. There is no suitable resolution pa-

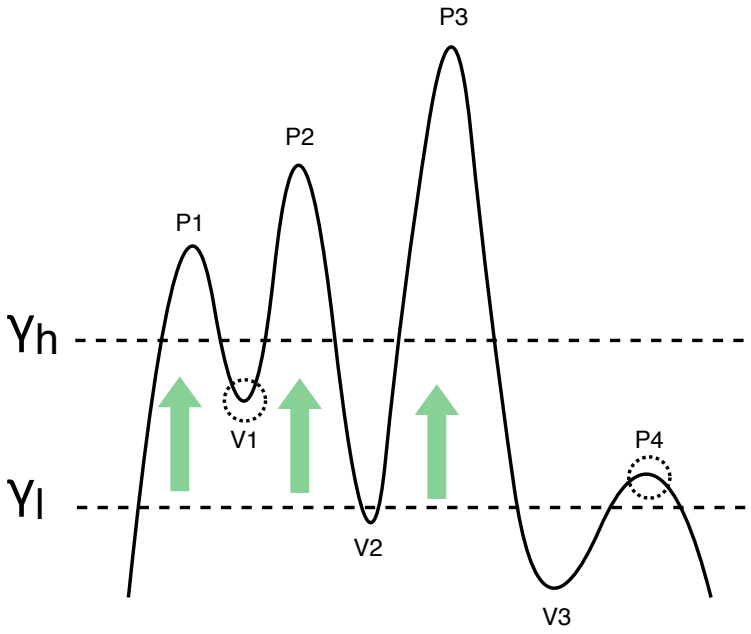


Fig. 2.3. The plateaus problem is analogous to finding mountains that are located at different plateaus; using a single altitude either would miss the lower mountains, or would treat the higher peaks as one mountain. Specifically, when using resolution parameter γ_l, the left two high peaks P1 and P2 are considered one "mountain" - two well-formed dense communities get merged because their inter-community edge density (illustrated by valley V1) is higher than γ_l. If we adopt a higher resolution parameter γ_h, the low peak P4 on the right gets ignored - a loose community gets split into multiple smaller clusters. Notably, this issue cannot be avoided as long as the valley V1 of the left two peaks P1, P2 is higher than the height of the right-most peak P4.

rameter γ to detect three communities in this case because the *density* parameter for the edges between two cliques 1.93 is larger than the *density* parameter for the edges inside random graph 1.03. When applying generalized modularity maximization, adopting a resolution parameter larger than 1.93, makes it likely that two cliques will be detected, but the random graph will get split into smaller communities. On the other hand, a resolution parameter within $[1.03, 1.93]$ preserves the random graph as one complete community, but the two clique gets merged into one community.

The issue is analogous to finding mountains that are located at different plateaus as shown in Fig. 2.3; using a single altitude either would miss the lower mountains, or would treat the higher peaks as one mountain. Specifically, when using resolution parameter γ_l, the left two high peaks in Fig. 2.3 are considered one "mountain" - two well-formed dense communities get merged. If we adopt a resolution parameter γ_h, the low peak on the right

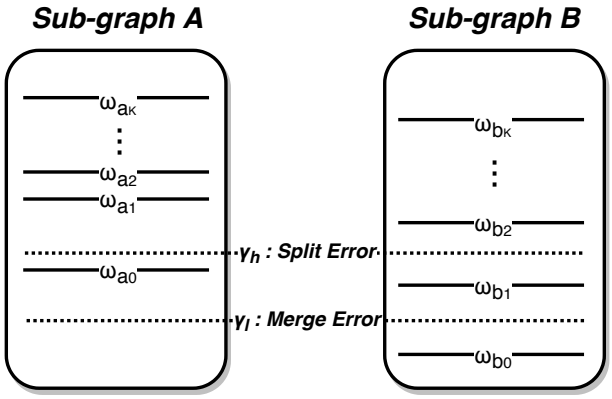


Fig. 2.4. Resolution limits of the generalized modularity can be explained by the relations between the values of the density parameters of stochastic block models. Given two disjoint subgraphs A and B such that the inter-community edge density ω_{a0} in subgraph A is larger than the intra-community edge density of some community in subgraph B, no suitable resolution parameter γ exists because Split Error and Merge Error cannot be resolved at the same time. Split Error occurs when the resolution parameter γ_h is larger than the inter-community edge density of a subgraph A, because the community b_1 with the intra-community density smaller than γ_h will be spread among multiple clusters; Merge Error occurs when the resolution parameter γ_l is smaller than ω_{a0} so the communities in subgraph A will be merged into one community.

gets ignored - a loose community gets split into multiple smaller clusters. Notably, this issue cannot be avoided as long as the valley of the left two peaks is higher than the height of the right-most peak.

More formally, given the *density* matrix of a degree-corrected stochastic block model and a set of communities $S = \{r\}$, the sub-matrix $\Omega_{S,S}$ formed by the rows and columns in $r \in S$ corresponds to a subgraph in the networks. Suppose subgraphs A and B have the inter-communities *density* parameter $\omega_{w_{a0}}$ and $\omega_{w_{b0}}$ respectively. In Fig. 2.4, using γ_h causes *Split Error* which splits some community with $\omega_{b1} < \gamma_h$ in B while using γ_l causes *Merge Error* which merges all communities in subgraph A.

This problem is more common in large networks than in small ones, as large networks are more likely to have inhomogeneous subgraphs in different regions. For this reason, a uniform resolution limit parameter is not sufficient to resolve communities located at different "plateaus". Motivated by this "plateaus" phenomenon, we propose a multi-scale community detection algorithm which gradually increases the resolution parameter to detect community in local subgraphs.

We propose an agglomerative heuristic algorithm which recursively divides the network into subgraphs to detect communities at different scales. At each level of recursion, the algorithm applies a resolution parameter $\gamma < 1$ in attempt to avoid inappropriately splitting of loose communities. But it is likely to merge inappropriately small well-formed communities into large ones. Therefore, the discovered subgraphs are then passed to the next level of recursion to further detect communities with higher edge *density* parameter. This idea is illustrated in Fig. 2.3 where the peaks located at a higher plateaus need a high altitude γ_h for each to have its own community.

The remaining challenge is to determine when to terminate the recursion. As the network breaks into smaller subgraphs recursively, the algorithm should stop when there is actually only one community in each subgraph. Indeed, one can always increase the resolution parameter to detect higher resolution communities in this subgraph. But it does not mean the current subgraph always contains community structures. For instance, a Erdos-Renyi random graph [41] can be partitioned into communities as long as the resolution parameter is high enough. However, we do not claim that Erdos-Renyi graph has community structures.

To ensure the detected communities are meaningful, we apply a **hypothesis testing framework** to ensure the significance of the partition. The *null* model H_0 here is defined as a simplified version of the degree-corrected planted partition model in which only one community exists, and the more general, nesting alternative H_1 is defined as the degree-corrected planted partition model whose log-likelihood is represented by Eq. 2.17 given the current partition $\mathbf{g}$.

The *null* model H_0 here is a special case of the degree-corrected planted partition model [26]. The number of edges between nodes i and j follows the Poisson distribution with mean

$$\eta_{ij} = \omega \frac{k_i k_j}{2m}, \tag{2.49}$$

where ω is a model parameter independent of the block assignments. Hence, we can consider this model as the degree-corrected planted partition model with only one community and the ω is the corresponding *density* parameter. The log-likelihood of this *null* model can be

simplified from Eq. 2.9 to the following

$$\begin{aligned}\log P_{\text{null}}(\mathbf{A}|\omega) &= \frac{1}{2}\sum_{ij}\left[A_{ij}\log\left(\omega\frac{k_i k_j}{2m}\right) - \omega\frac{k_i k_j}{2m}\right] \\ &= m\left(\log w - w\right) + \sum_i k_i \log k_i - m\log\left(2m\right).\end{aligned} \tag{2.50}$$

Taking the first-order derivative of the log-likelihood above over ω, it is easy to obtain the maximum likelihood estimate $\hat{\omega} = 1$, hence the posterior log-likelihood can be written as

$$\log P_{\text{null}}(\mathbf{A}) = -m + D, \tag{2.51}$$

where D is the constant term given in Eq. 2.18, which actually can be cancelled eventually in the test statistic as shown below.

It is worth noting that the *null* model defined here is similar to the configuration model [34], but they are not exactly equivalent. The configuration model [34] is a random graph model which assumes the edges are placed randomly between the nodes, while the degree of every node after such randomization is equal to the corresponding value in the original network. The network generation process in the configuration model can be understood as follows: The degrees of the vertices are represented as the number of half-links or stubs. These stubs are randomly paired with each other to create the edges. Hence, the configuration model produces an ensemble of graphs with the exact degree sequence as in the original network. The number of edges between nodes i and j averaged over the ensemble of graphs generated in this way is equal to $\frac{k_i k_j}{2m}$ where m is again the total number of edges in the original network and k_l is the degree of node l. In the *null* model defined here, when the ω parameter takes the MLE value $\hat{\omega} = 1$, the expected number of edges between nodes i and j is also $\frac{k_i k_j}{2m}$. But this *null* model allows multi-edges and self-edges (edges connecting a node to itself). The expected degree of the node i in the *null* model is $\sum_j \frac{k_i k_j}{2m} = k_i \frac{\sum_j k_j}{2m} = k_i$.

In general, the alternative model H_1 splitting a network into multiple communities should fit better to the observed network than the *null* model does because the alternative model involves many more model parameters. The log-likelihood of the degree-corrected planted partition model, $\log P_{\text{pp}}$, is supposed to be higher than the log-likelihood of the *null* model, $\log P_{\text{null}}$ as defined in Eq. 2.17. Therefore, we use the log-likelihood ratio statistic (LLR) [42] as a test statistic to measure their difference. Given a partition of the network

$\mathbf{g}$, the LLR is written as

$$\begin{aligned}\Lambda_{\mathbf{g}} &= -2\log\frac{P_{\text{null}}(\mathbf{A})}{P_{\text{pp}}(\mathbf{A};\hat{\Omega}_{\mathbf{g}},\mathbf{g})} \\ &= 2\left[\log P_{\text{pp}}\left(\mathbf{A};\hat{\Omega}_{\mathbf{g}},\mathbf{g}\right) - \log P_{\text{null}}\left(\mathbf{A}\right)\right],\end{aligned} \tag{2.52}$$

where $\hat{\Omega}$ is the posterior most-likely *density* parameters estimated from the given partition $\mathbf{g}$. The log-likelihood ratio statistic is equal to twice the difference between the log-likelihood of two models, $\log P_{\text{pp}}$ and $\log P_{\text{null}}$.

It is worth noting that the partition $\mathbf{g}$ is detected by the generalized modularity maximization and used to compute $\log P_{\text{pp}}(\mathbf{A};\hat{\Omega}_{\mathbf{g}},\mathbf{g})$ in our case. A low resolution parameter γ value is used for the generalized modularity maximization here because it is likely to result in a small number of communities, limiting the number of parameters in the alternative model H_1. Hence, it prevents the H_1 model from overfitting. Most importantly, this choice avoids the multiple re-estimation of γ, which is computationally expensive because it needs to maximize the generalized modularity over the same network many times with different γ values [26]. In practice, we choose a γ value slightly smaller than 1 in the experiments.

Plugging the specific expression of the log-likelihoods of these two competing models, Eq. 2.17 and Eq. 2.51, into Eq. 2.52 yields the test statistic in a simple form

$$\begin{aligned}\Lambda_{\mathbf{g}} &= 2\left[B\cdot Q\left(\hat{\gamma},\mathbf{g}\right) + C + m\right] \\ &= m\left[\log\frac{\hat{\omega}_1}{\hat{\omega}_0}\cdot Q\left(\hat{\gamma},\mathbf{g}\right) + \left(\log\hat{\omega}_0 - \hat{\omega}_0\right) + 1\right],\end{aligned} \tag{2.53}$$

where the constants B and C are defined in Eq. 2.18 but the parameters ω_1 and ω_0 take their MLE values, and $Q(\hat{\gamma},\mathbf{g})$ is the generalized modularity of the partition $\mathbf{g}$ with a resolution parameter $\hat{\gamma}$ defined by posterior maximum likelihood estimate, $\hat{\omega}_0$ and $\hat{\omega}_1$ are the *density* parameters obtained by the posterior maximum likelihood estimation given partition $\mathbf{g}$. With $\hat{\gamma}$ and $\mathbf{g}$, it takes $O(m)$ time to compute $Q(\hat{\gamma},\mathbf{g})$ where m is the total number of edges in the network. Given the modularity $Q(\hat{\gamma},\mathbf{g})$, the LLR test statistic can be computed in constant time.

In general, the alternative model H_1 splitting a network into multiple communities should perform at least as well as the *null* model, because the alternative model involves many more model parameters, i.e. the *density* matrix Ω. H_1 is accepted when the fit is

significantly better, i.e. $\Lambda_{\mathbf{g}}$ is large enough. Such significance is measured by the *p-value* of $\Lambda_{\mathbf{g}}$ defined as

$$p\text{-}value = P[\Lambda_{\text{null}} > \Lambda_{\mathbf{g}}], \tag{2.54}$$

where Λ_{null} is the corresponding log-likelihood value under the *null* hypothesis, which can be computed numerically by sampling a series of *null* networks generated by the *null* model. If the *p-value* is smaller than the significance level, the *null* hypothesis is rejected and the algorithm does hypothesis testing on the resulting communities; Otherwise, the algorithm returns the current subgraph as one single community and stops the current recursion branch. The hypothesis testing procedure is summarized below:

- Maximize the generalized modularity with a predefined $\gamma < 1$ to obtain partition $\mathbf{g}$.
- Estimate the *density* parameters $\hat{\Omega} = \{\hat{\omega}_0, \hat{\omega}_1\}$ of degree-corrected planted partition model by Eq. 2.21 and the corresponding resolution parameter $\hat{\gamma}$ by Eq. 2.19. Then compute the generalized modularity $Q(\hat{\gamma}, \mathbf{g})$ so the log-likelihood ratio test $\Lambda_{\mathbf{g}}$ can be obtained.
- Generate a series of *null* networks via the *null* model. Compute Λ_{null} for each *null* network, output the fraction of Λ_{null}s which are larger than $\Lambda_{\mathbf{g}}$ as the *p-value*.
- If the *p-value* is smaller than the significance level, reject H_0 and continue partitioning the subgraph. Otherwise, accept H_0 and return the current subgraph as a community.

It is worth noting that, according to the Wilks' theorem [43], when the sample size approaches infinity, the log-likelihood ratio test statistic is asymptotically chi-squared distributed when the null hypothesis holds true. However, the Wilks' theorem does not apply here for the test statistic as defined in the form of Eq. 2.52. To compute the *null* distribution, we need to enumerate a large set of *null* networks to calculate the *p-value*. In practice, the quality of $\mathbf{g}$ also influences the test statistic Λ_g. We observed that modularity maximization over a larger network often detects better $\mathbf{g}$ than over smaller ones, resulting in large test statistic Λ_g. Therefore, we find it more computationally efficient to terminate the current recursion branch when $\Lambda_{\mathbf{g}} < \tau * n_{sub}$ where τ is a constant value and n_{sub} is the number of nodes in the currently considered subgraph. The accurate calculation of *p-value* by sampling *null* networks can still be used for relatively small network when needed.

To evaluate the performance of the proposed multi-scale community detection algorithm, we compare it with the state-of-art greedy modularity maximization algorithm, Fast Greedy [5], on several real and synthetic networks. For the networks with pre-defined ground truth communities, the quality of the detected communities are evaluated by the Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) metrics which requires ground truth communities. Besides, we also measure the distribution of the community sizes which usually reflects the resolution limits problem because modularity maximization either combines smaller well-formed communities into bigger ones or splits larger well-formed communities into smaller ones. The definition of the two quality metrics mentioned above are given below.

One of the standard sources of community structures for the evaluation of community detection algorithms is the LFR benchmark [44] which generates networks based on a set of pre-defined ground truth communities. In so generated networks, both the degree and community size distributions follow the power law. The main benefit of using LFR benchmark is that the ground truth communities are known. The generated networks vary with the following three parameters: γ which is an exponent of the node degree in the power law distribution, β which is an exponent of the community size in the power law distribution, and μ which is the *density* parameter that defines the fraction of all edges which have both endpoints inside the same community.

In our experiments, the networks generated by the LFR benchmarks have the average node degree of 9.3 and the numbers of nodes ranging from 6,000 to 11,000. The exponents γ and β are set to 3.0 and 2.0 respectively and the *density* parameter μ is equal to 0.25. We evaluate the modularity maximization performance using the Fast Greedy algorithm [5]. The results are measured by the NMI and ARI metrics as shown in Table 2.2. The number of communities as a function of their sizes is plotted in Fig. 2.5. One notable difference between the community detection results is that the modularity maximization merges smaller communities into larger ones so there is fewer small communities in the results than in the ground truth. This is the main reason why modularity maximization does not perform as well as the proposed multi-scale community detection which recursively divides a large community into small ones until the probability that it contains communities becomes statistically insignificant.

The American college football network [45] consists of 115 nodes representing college

Table 2.2. Performance of multi-scale community detection compared to the Fast Greedy [5] modularity maximization algorithm on the LFR benchmark networks. Note large increase in value of metrics for multi-scale algorithm, at least 38% for NMI and 240% for ARI.

#Nodes	#Edges	Metric	Fast Greedy	Multi-scale
5000	23436	ARI	0.20368	0.69378
		NMI	0.60266	0.83706
7000	31546	ARI	0.12377	0.71148
		NMI	0.62044	0.87240
9000	41595	ARI	0.11038	0.74156
		NMI	0.59043	0.87355
11000	51430	ARI	0.12701	0.75043
		NMI	0.56224	0.86722

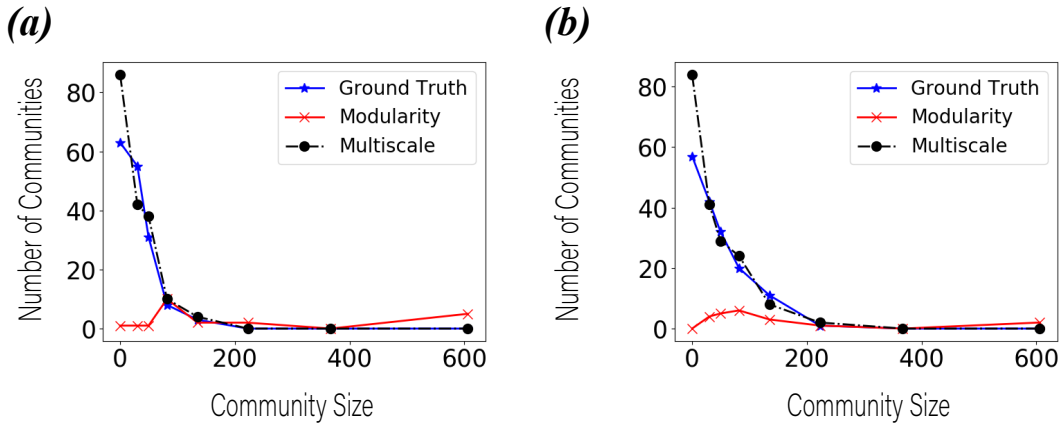


Fig. 2.5. Histogram of detected community sizes using the state-of-the-art modularity maximization algorithm, Fast Greedy [5], and the proposed multi-scale modularity maximization approach. This approach detects fewer small communities and more large communities compared to the ground truth due to the resolution limit problem. In contrast, the multi-scale approach detects the numbers of communities of over wider range of sizes. (a) LFR benchmark network with 7000 nodes (b) LFR benchmark network with 11,000 nodes.

football teams playing in a league with 11 conferences. Every edge in the American college football network denotes the positive number of games played by two teams in the year 2000 season. According to [45], each of the 11 college football conferences active at the time gets identified as one community because teams within a conference play more frequently with each other than with teams from other conferences. There are 8 independent teams (not

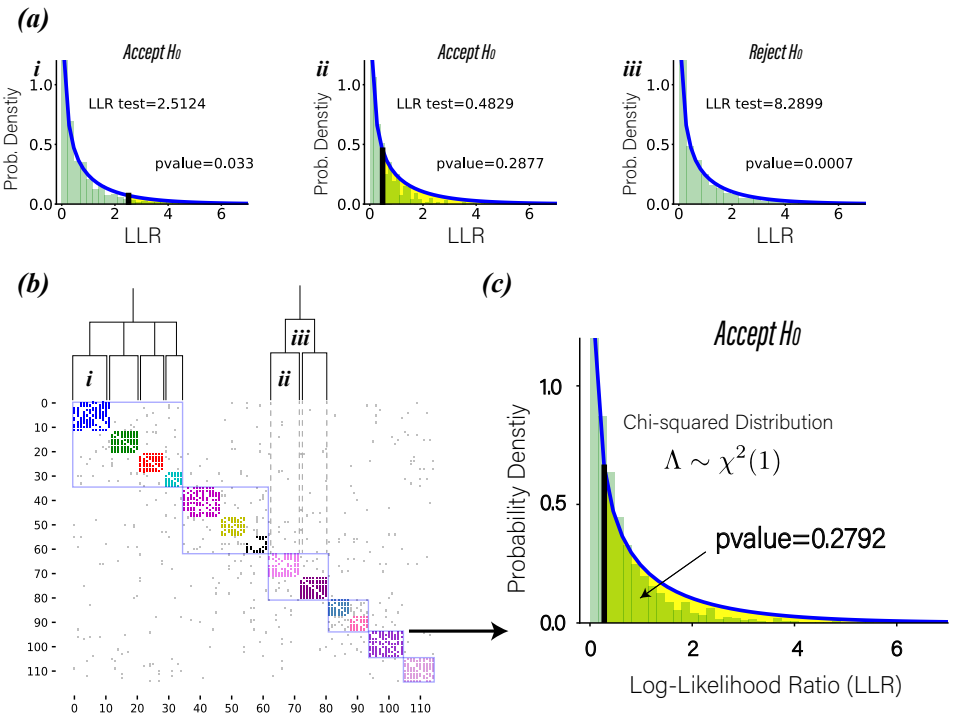


Fig. 2.6. Illustration of the multi-scale community detection in the American college football network [45]. (a) Three different hypothesis testing cases are: in cases (i) and (ii) the *null* hypothesis is accepted because statistically significant community has been found while in case (iii) the *null* hypothesis is rejected because the subgraph actually consists of two smaller communities. (b) The edge matrix where each dot indicates the one edge. Different colors define final communities detected by multi-scale community detection, while the rectangles show the communities detected at the first level. (c) A well-formed community passes the statistical significance test at the first level, and thus avoids further splitting. The green histogram shows the distribution by sampling *null* networks, the blue curve indicate the chi-squared distribution, and the size of the yellow area corresponds to the *p-value*.

members of any conference), each forming a single community.

Fig. 2.6 illustrates the steps of significance testing of the communities at two different levels. The generalized modularity with a predefined resolution parameter $\gamma = 0.9$ is maximized to obtain partitions. As shown in Fig. 2.6(b), at the first level, the networks is partitioned into 6 communities covered by the rectangles - some correspond to the well-formed ones because their community *densities* are larger than the resolution bound; the others, however, consist of multiple well-formed but smaller communities, each represented

by an unique color. Then, the algorithm does hypothesis testing given the partition at the first level - it estimates the corresponding posterior maximum-likelihood estimates of *density* parameters $\hat{\Omega}$ and evaluate the log-likelihood ratio test $\Lambda_{\mathbf{g}}$. Then, the algorithm generates a series of *null* networks via the configuration model, given the degree sequence of the network. Since the *p-value* is smaller than the significance level 0.01 chosen here, which is typically used for such tests, we reject the *null* hypothesis H_0 and continue partitioning the each of the 6 communities. At the second level, each individual community is treated as a subgraph, and the same generalized modularity maximization procedure is repeated on each of them again. The second community at the bottom right corner has a high community *density*, thus, the *null* hypothesis gets accepted with high p-value of 0.2792, in case (i) and similarly in case (ii). However, in case (iii), as shown in Fig. 2.6(a)(iii), the *p-value* 0.0007 is much smaller than the significance level of 0.01 used here. Therefore, the multi-scale community detection algorithm rejects H_0 and further splits that ill-formed community into two smaller ones. The modularity maximization conducted by Fast Greedy obtains 6 communities similar to the ones covered by the six rectangles in Fig. 2.6(c) and it obtains a NMI score of 0.5572. The proposed multi-scale community detection algorithm finds 13 communities, achieving the NMI scores of 0.8728.

2.5 Regularized Stochastic Block Model

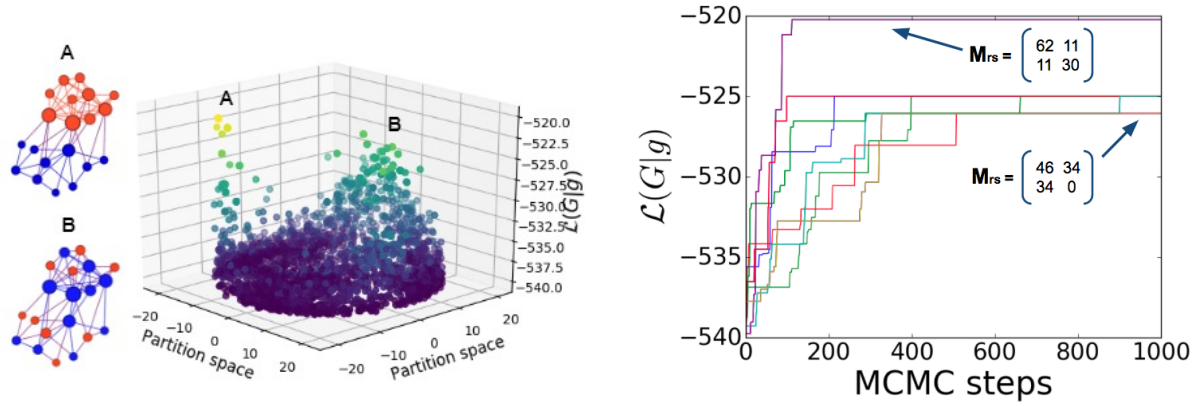
As defined in previous literatures [40], [46], the assortative structures correspond to the traditional community structures, nodes are more frequently connected with each other inside communities than with the rest of the network. The disasortative structures do not satisfy this condition. For example, the core-periphery structures [47] divides the networks nodes into the core part, where nodes are often the hubs of the networks, and the periphery part, where nodes of low-degree connect to the core nodes. This is an advantage of the degree-corrected stochastic block model because it allows the discovery of the structures with different mixing patterns. However, when searching for the weak assortative community structures, the inference algorithm may still return the disassortative structure [48] instead.

We generate synthetic networks using the degree-corrected stochastic block model with a block assignment $\{g_i\}$ and the parameters ω_{rs} chosen for the assortative communities as

follows

$$\omega_{rs} = \begin{cases} \gamma\omega_0 & \text{if } r = s, \\ \omega_0 & \text{if } r \neq s. \end{cases} \tag{2.55}$$

where a large value of γ results in strong community structure while ω_0 controls the sparsity of the network. The degree sequence is drawn from a power-law distribution with exponent 2.5, and the block assignments are randomly assigned, but each block is of the same size.



(a) Local optima of log-likelihood

(b) Convergence

Fig. 2.7. The convergence of 20 Markov Chain Monte Carlo (MCMC) trials for the degree-corrected stochastic block model. The local optimum A found by MCMC represents the assortative communities, whereas there are other local optima representing disassortative structures such as point B. (a) Multiple locally optimal partitions discovered by the MCMC inference for degree-corrected stochastic block model; (b) 2 out of the 20 MCMC trials find the most likely and sought-after assortative partition, while the other trials get trapped at the local optima. The matrix M_{rs} indicates the number of edges between every pair of blocks.

Given the synthetic networks produced by the degree-corrected stochastic block model, we infer the block assignment $\{g_i\}$, recovering the parameters used for its generation. Specifically, the number of communities is set as two for both the generation and recovery. To generate samples, the degree-corrected stochastic block model uses $\omega_0 = 0.01$ and $\gamma = 10$. Each of the two blocks contains 10 nodes. We scatter the sampled partitions on the x-y plane in Figure 2.7 and the z-axis indicates the log-likelihood of the corresponding partitions. We adapt the Markov chain Monte Carlo (MCMC) algorithm [49], [50] to infer the block assignment of the stochastic block model and its extensions. The MCMC algorithm [50] samples

the distribution of all possible block assignments in the network partitions space. For each trail, the algorithm starts from a random partition and iteratively, with certain probability, traverse to an adjacent partition. An entire MCMC sweep of all nodes in the network requires $O(E)$ operations where E is the number of edges in the networks [50]. The mixing time of the MCMC algorithm depends on the specific input network and the starting point. The details of the MCMC algorithm are discussed in the Supplementary Information.

Figure 2.7 shows the existence of multiple local optima in the log-likelihood of degree-corrected stochastic block model. The inference process finds three local optima here: (i) partition A which corresponds to the assortative structure matching the ground truth block assignment used for its generation; (ii) partition B which corresponds to a disassortative structure; and (iii) another disassortative partition which is not explicitly marked in the Figure 2.7(a). Under the degree-corrected stochastic block model, the MCMC inference starting from random initial partition finds the most suitable partition A in only 2 out of 20 trials, whereas the other attempts are trapped at the local optima as shown in Figure 2.7(b).

Since there are multiple local optima in the log-likelihood of the degree-corrected stochastic block model, the inference algorithm may converge to any of them nondeterministically. Specifically, the type of the discovered structure depends on the trial starting point and inference algorithm parameters. To avoid such nondeterministic outcomes, we introduce a novel approach called Regularized Stochastic Block Model (RSBM) applicable to any inference algorithm. RSBM constrains nodes' internal degree ratios, each of which is defined in the objective function as the fraction of a node's neighbors inside the same community. The resulting algorithm reliably finds assortative or disassortative structures as directed by the value of a single parameter.

We extend the formulation of the expected number of edges between nodes i and j, determined by the *Poisson* rate λ_{ij}, in the degree-corrected stochastic block model by defining it as

$$\lambda_{ij} = \begin{cases} \omega_{g_i,g_j} I_i I_j & \text{if} \quad g_i = g_j \\ \omega_{g_i,g_j} O_i O_j & \text{otherwise} \end{cases} \tag{2.56}$$

where any node l has two associated parameters I_l and O_l. Given Eq. 2.9, the log-likelihood

of generating graph G by this regularized stochastic block model can be written as

$$\mathcal{L}(G|\mathbf{g},\omega,\mathbf{I},\mathbf{O}) = 2\sum_i \left(k_i^+ \log I_i + k_i^- \log O_i\right) + \sum_{rs} m_{rs} \log \omega_{rs} - \omega_{rs}\Lambda_{rs} \tag{2.57}$$

where k_i^+ is the number of neighbors of node i which are inside the same block given the block assignment $\mathbf{g}$ and $k_i^- = k_i - k_i^+$. Thus, we get

$$\Lambda_{rs} = \begin{cases} (\sum_{i\in r} I_i)^2 & \text{if } r = s \\ \sum_{i\in r} O_i \sum_{i\in s} O_i & \text{if } r \neq s \end{cases} \tag{2.58}$$

To simplify, we write $i \in r$ if $g_i = r$. For block assignment $\mathbf{g}$, the maximum-likelihood values of ω_{rs} are

$$\hat{\omega}_{rs} = \frac{m_{rs}}{\Lambda_{rs}} \tag{2.59}$$

Dropping the constants and substituting using Eq. 2.57, we obtain

$$\mathcal{L}(A|\mathbf{g},\mathbf{I},\mathbf{O}) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{\Lambda_{rs}} + 2\sum_i \left(k_i^+ \log I_i + k_i^- \log O_i\right) \tag{2.60}$$

Note that if we set $I_i = O_i = 1$ here, the log-likelihood above reduces to the definition of standard stochastic block model with $\Lambda_{rs} = n_r n_s$ which is exactly the product of the sizes of two blocks r and s. When $I_i = O_i = k_i$, the second sum on the right hand side (RHS) becomes irrelevant to the maximum likelihood estimation (MLE) result. Hence, the log-likelihood reduces to the definition of degree-corrected stochastic block model in Eq. 2.60 with $\Lambda_{rs} = \kappa_r \kappa_s$, i.e., the product of the sums of degrees of nodes in two blocks r and s. Hence, by introducing two sets of parameters $\mathbf{I} = \{I_i\}$ and $\mathbf{O} = \{O_i\}$ in the edge probability, we obtain a more generalized definition of stochastic block model here.

For alternative formulation of our model, we define the prior in-degree ratio $f_i = I_i/(I_i + O_i)$ and $\theta_i = I_i + O_i$ for each node i. By rewriting the second summation on the RHS of Eq. 2.60, we get

$$\mathcal{L}(G|\mathbf{g},\mathbf{I},\mathbf{O}) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{\Lambda_{rs}} - 2\sum_i k_i H(\frac{k_i^+}{k_i}, f_i) + 2\sum_i k_i \log \theta_i \tag{2.61}$$

where $H(\frac{k_i^+}{k_i}, f_i) = -\frac{k_i^+}{k_i} log f_i - \frac{k_i^-}{k_i} log(1-f_i)$ represents the cross entropy between the observed

and prior in-degree ratio. Therefore, the prior in-degree ratios $\{f_i\}$ regularizes the in-degree ratios $\{\frac{k_i^+}{k_i}\}$ in the resulting partition by maximizing Eq. 2.61.

In real networks, the low degree nodes are more likely to have neighbors inside a block than the high degree nodes are. Suppose f_i depends only on the degree of node i, i.e. $f_i = f(k_i)$. Then, the function $f(k) : \mathbb{Z}_+ \to [0, 1]$ should be strictly decreasing. In an assortative partition of the network, we have

- $f(1) = 1$ because a node with degree one must connect to the community it belongs to;
- for $k \approx |V|$, $f(k) \ll 1$ because a super-hub eventually does not belong to any community as its degree is of the order of the number of nodes in the entire network.

A simple function $\{f(k)\}$ satisfying this requirement is of the form

$$f(k) = \alpha + \frac{(1-\alpha)}{k} \tag{2.62}$$

where α is the only extra parameter we introduce to the regularized stochastic block model (RSBM). Alternatively, we can select a constant $f \in (0, 1)$ such that

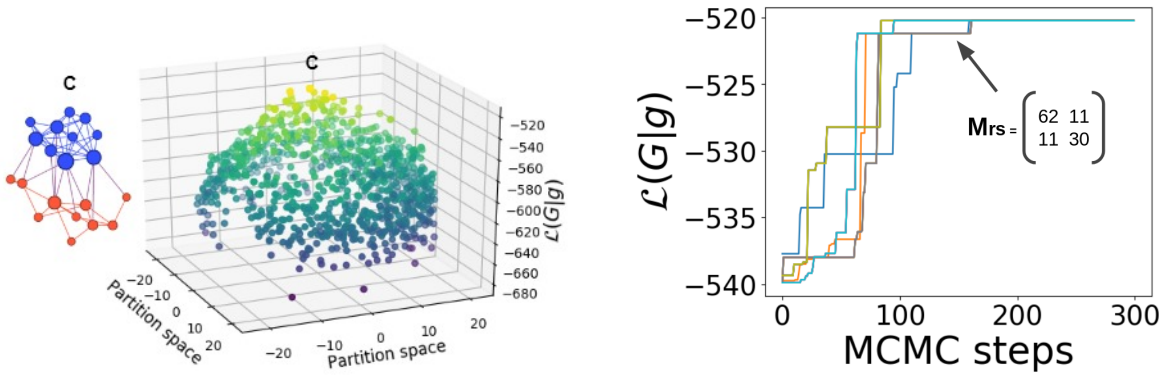
$$f(k) = max(f, \frac{1}{k}) \tag{2.63}$$

The impact of different choices of f_i on the discovered block assignment is discussed in the following two subsections presenting experimental results. It is worth noting that, for either choice of the prior in-degree ratio $\{f_i\}$, there is only one extra parameter, i.e. α or f, introduced here in addition to the original parameters of the degree-corrected stochastic block model.

We generate synthetic networks with the assortative communities using the degree-corrected stochastic block model. The parameters ω_{rs} in the network generation process are specified by Eq. 2.55. By selecting a small value of $\gamma > 1$ in Eq. 2.55, the generated community structures are relatively weak, thus, it is more difficult to detect them using the statistical inference of the degree-corrected stochastic block model.

We adapt the Markov chain Monte Carlo (MCMC) algorithm [49] to infer the block assignment of the stochastic block model and its extensions. Figure 2.8(a) shows there is only one unique local optimum, the partition C, found by 20 MCMC trials under the

regularized stochastic block model. Therefore, all 20 MCMC trials converge to this unique local optimum. As shown in Figure 2.8(b), the Markov Chain Monte Carlo inference finds the correct block assignment within only 150 steps. The regularization terms made it possible to avoid the unsuitable local optima during the inference.



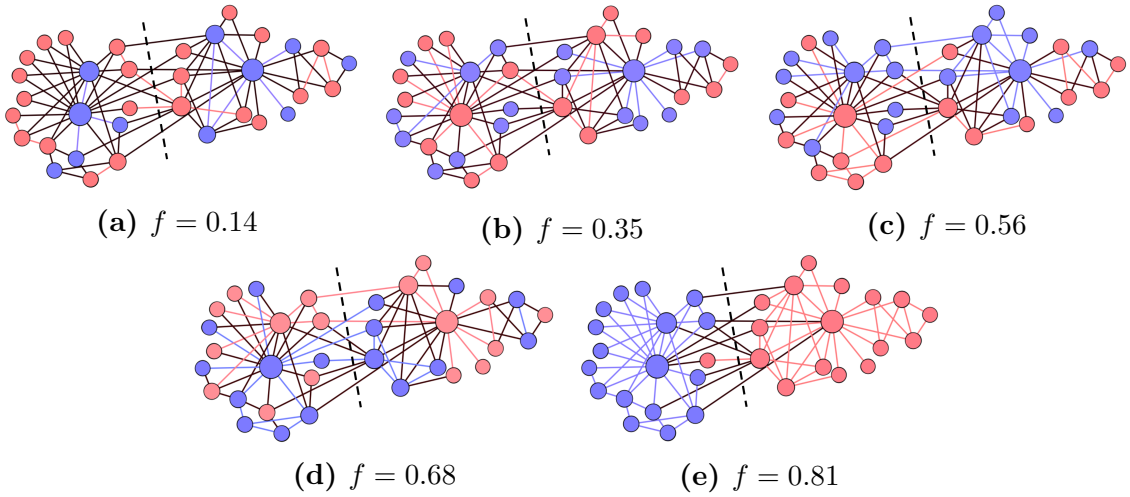
(a) Local optima of log-likelihood

(b) Convergence

Fig. 2.8. The convergence of 20 Markov Chain Monte Carlo (MCMC) trials for the regularized stochastic block model (RSBM) introduced here. All trials converge to the local optimum C which represents an assortative structure. (a) One local optimal partition observed during the MCMC inference for our model. (b) All twenty MCMC trials find the sought-after assortative structure.

We use the networks including Karate club network of Zachary [38], the Dolphin social network of Lusseau et al. [39] and the network of fictional characters' interactions in the novel Les Miserables by Victor Hugo [40] to demonstrate the performance of the regularized model introduced here. The details of each network are presented in the Supplementary Materials. For Karate club network, we evaluate the effect of the regularization terms on the resulting partitions using Markov chain Monte Carlo (MCMC) as the inference algorithm. For every node i in the network, we set the parameter $\theta_i = k_i$ and $f_i = max(f, 1/k_i)$ for our regularized stochastic block model defined by Eq. 2.61 where k_i is the degree of node i and f_i represents the prior in-degree ratio for regularization. Figure 2.9 shows the most likely partition of the Karate club network found by MCMC using different f values. The color represents the block assignment, and the black dashed line divides the network into two parts in the ground truth partition. As shown in Figure 2.9, when $f = 0.14$, the inference algorithm outputs a core-periphery structure which clusters high-degree nodes into the blue

block and the remaining low-degree nodes into the red block. This is because the sum of cross entropy terms serves as a regularization term which penalizes those partitions that assign adjacent nodes into the same block. As the value of f grows, the inference algorithm is more likely to detect assortative structure. When $f = 0.85$, the inferred block assignment matches the ground truth partition of the Karate club network with the exception of one single red node. However, this node has only one connection to each block; thus, it is quite arguable to which block this node should belong.



(a) $f = 0.14$ **(b)** $f = 0.35$ **(c)** $f = 0.56$

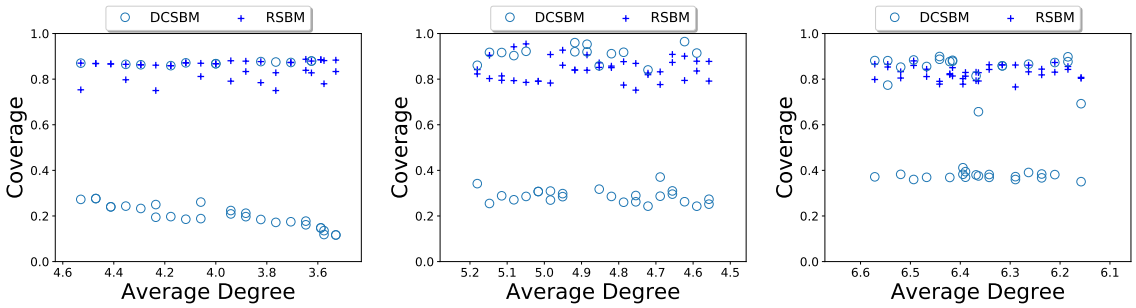
(d) $f = 0.68$ **(e)** $f = 0.81$

Fig. 2.9. The partition of Karate network inferred by Markov Chain Monte Carlo under different parameter settings where nodes of the same color belong to the same partition. The black dotted line represents the ideal partition in the ground truth. A small f results in the core-periphery partition of the network while a large f leads to assortative partitions. The values of the f parameter are shown in the corresponding sub-figures captions.

The results in Figure 2.10 show that the MCMC inference of our RSBM model on these networks finds partitions with higher *coverage* than the ones inferred by the degree-corrected stochastic block model. The *coverage* of a partition [46] is defined as the ratio of the number of edges with both endpoints in the same block to the total number of edges in the entire network

$$\text{coverage}(\mathbf{g}) = \frac{|\{(i,j) \in E | g_i = g_j\}|}{|E|} \tag{2.64}$$

where E is the set of edges in the network. A low *coverage* indicates that the resulting partition is disassortative. An ideal assortative partition of the network, where all clusters are disconnected, yields a *coverage* of 1.



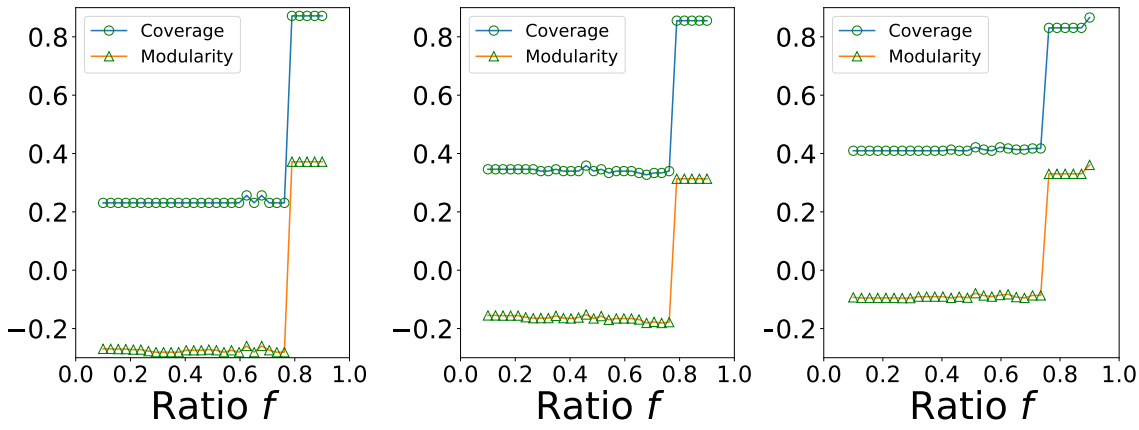
(a) Karate club **(b) Dolphin social** **(c) Les Miserables**

Fig. 2.10. The *coverage* of partitions in real networks as a function of average node degree. The optimal partitions are inferred by the Markov Chain Monte Carlo algorithm under degree-corrected stochastic block model (DCSBM), marked by circles, and its regularized extension (RSBM) introduce here, marked by crosses. (a) Karate club network [38]; (b) Dolphin social network [39]; (c) Characters interaction network of Les Miserables [40].

Figure 2.10 shows the *coverages* of the partitions found in the three real networks mentioned above. We randomly remove edges in these networks to further increase the sparsity of these networks. And the numbers of blocks used in trials are set to their values for these real networks well-accepted in the literature. The results in Figure 2.10 indicate that, under degree-corrected stochastic block model (DCSBM), the inference algorithm is likely to miss the assortative structures and returns instead the disassortative partitions of the network, which also fit the model in such cases. In contrast, the MCMC inference of our regularized stochastic block model (RSBM) almost deterministically produces the assortative structures. Interestingly, the inference of degree-corrected stochastic block model produces the partitions with the *coverage* values distributed at two levels. There is no partition with a *coverage* between these two levels found by the inference algorithm. This is similar to the case of the synthetic network in Figure 2.7 where both the assortative communities and disassortative structures fit the degree-corrected stochastic block model. Which structure is found is determined by random sampling of the potential partitions. In other words, there is no way to guide whether the disassortative structures or the assortative communities are preferred, so, two consecutive MCMC trials may return completely different structures.

In contrast, the inference of the introduced here regularized stochastic block model using the prior in-degree ratio $f_i = 0.8 + 0.2/k_i$ is very robust. It only produces partitions with a high *coverage*, which are generally distributed at the same level as the assortative

partitions under degree-corrected stochastic block model. Moreover, the sparsity of the networks does not have an obvious impact on the resulting partitions under our regularized model.



(a) Karate club **(b) Dolphin social** **(c) Les Miserables**

Fig. 2.11. The *coverage* and modularity of partitions in real networks as a function of parameter f. The optimal partitions are inferred by Markov Chain Monte Carlo algorithm under degree-corrected stochastic block model (DCSBM) and its regularized extension (RSBM) introduced here. The networks used for evaluation include (a) Karate club network [38]; (b) Dolphin social network [39]; (c) Characters interaction network of Les Miserables [40].

.

We evaluate the modularity [24] of resulting partitions of the real networks as a function of parameter f. A high modularity indicates the strong community structure. In our experiments, we use a constant f such that $f_i = f$ for each node i in the network with degree $k_i > 1/f$, otherwise $f_i = 1/k_i$. We start with a small f value and increase it in each iteration. The MCMC inference uses as a starting point the original network initially, and the network partition found in the previous iteration subsequently. Figure 2.11 shows that, as the value of f increases, in general both the modularity and the *coverage* grow. When f is close to 0, the value of f does not have any effect on the network partition. However, as f becomes larger, then at a certain threshold of about 0.75, the increase of f leads to a critical transition from disassortative partitions to assortative communities. Then, a larger f value does not further increase the modularity of the partition. These results indicate that, with one single parameter f, the Markov Chain Monte Carlo inference gains the flexibility to choose between assortative communities resulting from clustering dense modules of nodes together

or disassortative structure, such as the core-periphery structure from clustering high-degree and low-degree nodes into separate blocks. With a choice of high f, the inference algorithm is likely to detect assortative communities.

2.6 Edge weighting scheme

The modularity can be naturally extended to the networks with weighted edges by replacing the count of edges with the sum of their weights. Hence, the weighted modularity is defined as

$$Q^w(G^w, C) = \sum_{c_i \in C} \left[\frac{W_{c_i}^{in}}{W} - \left(\frac{W_{c_i}}{2W} \right)^2 \right] \tag{2.65}$$

where W is the sum of weights of edges in the entire graph, $W_{c_i}^{in}$ is the sum of weights of edges within community c_i, and the weight of a community is defined as $W_{c_i} = 2W_{c_i}^{in} + W_{c_i}^{out}$ where $W_{c_i}^{out}$ is the sum of weights of edges with exactly one endpoint inside c_i. The original definition of modularity is a special case of the weighted version when the weight of every edge is 1. Many algorithms including [5], [51]–[55] were proposed to discover communities in a network by maximizing the modularity. One interesting finding is that Newman's modularity measure is related to the broader family of spectral clustering methods [55]. There are two categories of spectral algorithms for maximizing modularity: one is based on the modularity matrix [24], [56], [57], the other is based on the Laplacian matrix of a network [55], [58]. The first greedy algorithm, Fast Greedy [5], iteratively merges communities in the network to maximize the modularity. Initially, every node is a single community. In every step, two communities joining of which results in the largest modularity among all partitions created by temporary merging one pair of communities are merged together. After $|V| - 1$ steps, there is a single community remaining in the network and there are a total of $|V|$ partitions, each generated by a single step. Then the algorithm outputs the partition with the largest modularity. The greedy algorithms solve the maximization problem efficiently, yet they suffer from the resolution limit problem. This problem is defined as an increase of modularity when small well-formed (or ground truth) communities are undesirably joined together into a large community. As pointed out in [9], [59], this problem arises because the definition of modularity does not penalize for the increase of the diameter in a community created by merging together smaller ones. In recent work [60], Chen et al. introduced a new quality metric, called *modularity density*, to limit such bias towards large communities. The

new metric is also shown to be able to handle another known weakness of modularity, the counterproductive splitting of large communities. This is because the modularity density takes into account the density of discovered communities and penalizes the splitting of large communities. The fine-tuned Q_{ds} algorithm [6] was proposed to maximize this new quality metric.

2.6.1 Properties of balanced edge weighting scheme

Instead of proposing a new community detection algorithm, we invent an efficient edge weighting scheme to improve the performance of the current state-of-art algorithms. As shown in [9], for the modularity maximization to be able to find a community c_i with $|E_i^{in}|$ edges inside, the following inequality must hold,

$$|E_i^{in}| \geq \sqrt{\frac{|E|}{2}}. \tag{2.66}$$

where E_i^{in} is the set of the edges inside community c_i, and E is the set of edges of the entire graph. In large networks with millions of edges, the number of edges in most communities is often smaller than this lower bound. In [61], it has been shown that edge weighting scheme is capable of decreasing such theoretical bound and enhancing community detection performance in practice. Inspired by this result, we define the edge weighting scheme that *enhances* particular community as follows.

Remark 1. An edge weighting scheme *enhances* a community c_i by sum of additional weights $e_i = \sum_{e \in E_{c_i}^{in}} (w_e - 1) > 0$ if $w_e \geq 1$ for $\forall e \in E_{c_i}^{in}$, and $w_e \leq 1$ for $\forall e \in E_{c_i}^{out}$ while $W_{c_i} = d_{c_i}$ holds. Such a scheme is a *balance enhancement* if both communities connected by the cross-community edge with decreased weight are enhanced.

The edge weighting scheme *enhances* a community c_i by increasing the weights of edges residing within this community with added total weight of $e_i > 0$, while reducing the weight of edges crossing to other communities by $2e_i$ to preserve the weight of the community W_{c_i} equal to d_{c_i}. It is worth noting that a balanced enhancement preserves the total weight of the original graph, which is $W = |E|$. Here, we show that such balanced weighting scheme is non-decreasing modularity operation on a graph.

Theorem 1. *$Q^w(G^w, C) \geq Q(G, C)$ if the weighting scheme is balanced.*

Proof: For any community c_i,

$$W_{c_i}^{in} = \sum_{e \in E_{c_i}^{in}} w_e \geq \sum_{e \in E_{c_i}^{in}} 1 = |E_{c_i}^{in}|. \tag{2.67}$$

Thus,

$$Q^w(G^w, C) = \sum_{c_i \in C} \left[\frac{W_{c_i}^{in}}{W} - \left(\frac{W_{c_i}}{2W} \right)^2 \right] \tag{2.68}$$

$$\geq \sum_{c_i \in C} \left[\frac{|E_{c_i}^{in}|}{|E|} - \left(\frac{d_{c_i}}{2|E|} \right)^2 \right] \tag{2.69}$$

$$= Q(G, C). \tag{2.70}$$

Although the edge weighting scheme which *enhances* one community in a partition always increases this community's modularity, it does not necessarily guarantee that such enhanced partition would maximize modularity in the weighted graph. Here, we define a notion of locally maximal partition and prove that the proper edge weighting scheme can preserve such property.

Remark 2. Modularity of a partition C is *locally maximal* if the modularity decreases upon splitting any community in C or joining any two communities in C.

Theorem 2. *$\Delta Q^w_{c_i,c_j} \leq \Delta Q_{c_i,c_j}$ if c_i, c_j are enhanced by the balanced edge weighting scheme.*

Proof: Without loss of generality, consider two ground truth communities c_i and c_j *enhanced* by the balanced edge weighting scheme. The change in modularity Q^w upon joining these two communities is,

$$\Delta Q^w_{c_i,c_j} = \frac{W_{c_i,c_j}}{W} - \frac{W_{c_i} W_{c_j}}{2W^2} = \frac{W_{c_i,c_j}}{|E|} - \frac{d_{c_i} d_{c_j}}{2|E|^2} \tag{2.71}$$

$$\leq \frac{|E_{c_i,c_j}|}{|E|} - \frac{d_{c_i} d_{c_j}}{2|E|^2} = \Delta Q_{c_i,c_j} \tag{2.72}$$

Theorem 3. *If community c with $d_c \leq \sqrt{8|E|}$, is enhanced by the balanced edge weighting scheme and split into communities c_i, c_j and $\Delta Q_{c_i,c_j} \geq 0$ then also $\Delta Q^w_{c_i,c_j} \geq 0$.*

Proof: Consider a community $c = c_i \cup c_j$ with $d_c \leq \sqrt{8|E|}$ *enhanced* by the balanced edge weighting scheme, where c_i and c_j are two non-empty communities, then $\Delta Q_{c_i,c_j} \geq 0$ by

assumption that modularity reaches local maximum for partition C^*. If $E_{c_i,c_j} = \emptyset$, then $W_{c_i,c_j} = 0$ and $\Delta Q_{c_i,c_j} = -\frac{d_{c_i} d_{c_j}}{2|E|^2} \geq 0$. This leads to either $d_{c_i} = 0$ or $d_{c_j} = 0$ which causes contradiction. Otherwise, if $|E_{c_i,c_j}| \geq 1$, then $W_{c_i,c_j} \geq 1$ because the edge weighting scheme assigns weight $w_e \geq 1$ to any edge $e \in E_{c_i,c_j}$. Since $W_{c_i} + W_{c_j} = W_c = d_c$, we have $W_{c_i} W_{c_j} \leq (\frac{d_c}{2})^2$. When community c splits into communities c_i and c_j, the change in modularity Q^w is,

$$-\Delta Q^w_{c_i,c_j} = \frac{W_{c_i} W_{c_j}}{2W^2} - \frac{W_{c_i,c_j}}{W} \leq \frac{(\frac{d_c}{2})^2}{2W^2} - \frac{1}{W} \tag{2.73}$$

$$= \frac{(d_c)^2 - 8W}{8W^2} = \frac{d_c^2 - 8|E|}{8|E|^2} \leq 0 \tag{2.74}$$

Note that the last inequality holds because of the condition $d_c \leq \sqrt{8|E|}$. From Theorems 2 and 3 it follows immediately that if partition C^* is locally maximal and each community c satisfies the condition $d_c \leq \sqrt{8|E|}$ and is enhanced by the balanced edge weighting scheme, the modularity of this partition is locally maximal also for the weighted graph G^w. Since, by Theorem 2, joining communities c_i and c_j makes change $\Delta Q^w_{c_i,c_j} \leq \Delta Q_{c_i,c_j}$, it is entirely possible that $\Delta Q^w_{c_i,c_j} \leq 0 < \Delta Q_{c_i,c_j}$. Thus, modularity maximization for the graph with the enhanced weights will avoid joining possibly well-formed communities c_i and c_j while the maximization on the original graph would join them. This example demonstrates that if the well-formed small communities are *enhanced*, then their chances of being detected will increase. This observation motivates us to propose a regression model for assigning weights to edges so that the real ground truth communities can be *enhanced* in a network.

2.6.2 Graph sampling and edge weights regression

We follow the same paradigm of the original modularity maximization to detect communities but seek to assign weights to the edges to improve the quality of results. To be precise, a regression model is developed to convert an unweighted graph G to a weighted graph G^w so that modularity maximization finds communities of better quality by maximizing modularity in G^w rather than in G. The regression model takes the local topological features of edges as input and outputs the weight of every edge. As illustrated in Figure 2.12, the proposed procedure is divided into the following three steps:

- **Artificial network construction** done to estimate the network parameters of the input graph and construct a similar artificial graph in which the ground truth commu-

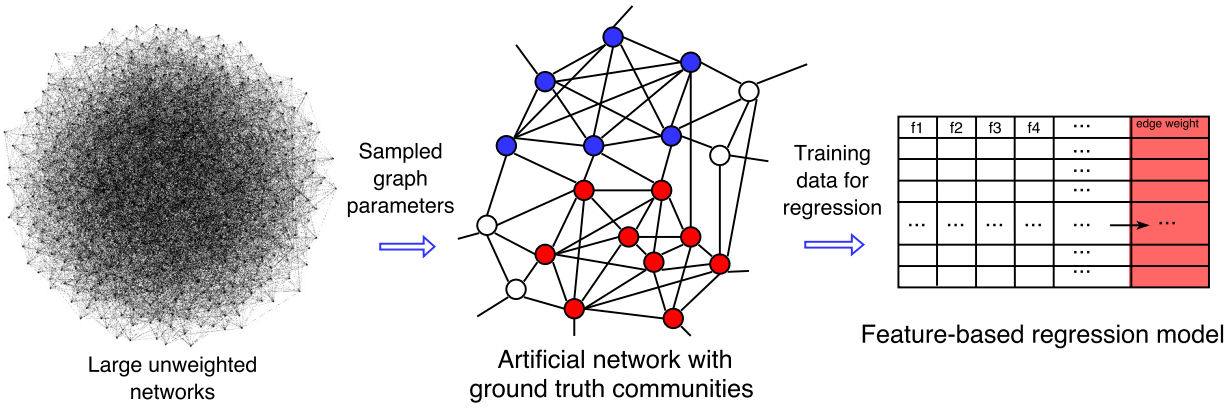


Fig. 2.12. Illustration of the adaptive modularity maximization.

nities are known beforehand by construction. The goal is to ensure these ground truth communities can be successfully separated in the modularity maximization process.

- **Edge feature extraction** executed on the edges of the artificial graph. The edge features are used as input to the regression model.

- **Regression on edge weights** uses a regression model to compute the edge weights such that the modularity maximization is able to separate adjacent ground truth communities in the artificial network.

Given a large unweighted input graph, our approach constructs an artificial network which shares degree distribution and clustering coefficients with the input graph. Specifically, multiple Stochastic Block Model (SBM) networks [62] are created with high intra-block edge densities and with a few randomly chosen inter-block edges, resulting in a relatively small inter-block edge density and with blocks forming ground truth communities. Then, the edges in these SBM graphs are randomly removed from network instances until the average node degree becomes close to that of the input graph. Among all SBM network instances, the one with the average clustering coefficient closest in its value to the input graph is chosen as the final artificial network. The ground truth communities (i.e., the nodes in blue and red in Figure 2.12) in the artificial network are used as training data to infer the parameters of the regression model. As Theorem 2 suggests, if the correct communities have been enhanced, then the probability of properly detecting these communities would increase. Therefore, the regression model incorporates these ground truth communities into the detection algorithm framework to enhance it. Since communities are considered local structures, the second step

of our approach is to extract the local topological features of every edge in the network. For each edge $e = (u, v)$ in the graph, the following local topological features are extracted efficiently from the network.

f-1. The square root of the number of common neighbors, $\sqrt{\mathcal{N}(u) \cap \mathcal{N}(v)}$, where $\mathcal{N}(v)$ denotes the set of neighbors of node v.

f-2. The difference in clustering coefficients of the endpoints, $|c(u) - c(v)|$, where $c(v)$ denotes the clustering coefficient of node v.

f-3. Jaccard-coefficient which is defined as

$$\text{Jaccard}(u, v) = \frac{|\mathcal{N}(u) \cap \mathcal{N}(v)|}{|\mathcal{N}(u) \cup \mathcal{N}(v)|} \tag{2.75}$$

f-4. Resource allocation index which is defined as

$$\bigcup_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \frac{1}{|\mathcal{N}(w)|} \tag{2.76}$$

f-5. Adamic-Adar index which is defined as

$$\bigcup_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \frac{1}{\log |\mathcal{N}(w)|} \tag{2.77}$$

f-6. Relative degree ratio which is defined as

$$\text{rel}(u, v) = \frac{\min(|\mathcal{N}(u)|, |\mathcal{N}(v)|)}{\max(|\mathcal{N}(u)|, |\mathcal{N}(v)|)} \tag{2.78}$$

When the degrees of nodes u,v are equal, $\text{rel}(u, v) = 1$. The attributes of edges or nodes, such as text content and user profiles, can also be used as features, if they are available. Using more local topological features generally leads to better accuracy because more information is embedded in these features. As pointed out in [9], community detection algorithms based on modularity maximization tend to execute counterproductive merges of small communities into large ones. One way to handle this resolution limit problem is to cause such merging operations to decrease the modularity. According to the definition of weighted modularity,

the change in Q upon joining two communities c_i and c_j is

$$\Delta Q^{w}_{c_i,c_j} = \frac{W_{c_i,c_j}}{W} - \frac{W_{c_i} W_{c_j}}{2W^2} \tag{2.79}$$

where W_{c_i,c_j} is the sum of weights of the edges between c_i and c_j, $W_{c_i} = 2W^{in}_{c_i} + W^{out}_{c_i}$ is twice the sum of weights of the edges inside community c_i plus the sum of weights of the edges with exactly one edge in c_i, and W is the sum of weights of all edges. To avoid the merging of some pairs of small communities $\{(c_i^1, c_i^2)\}_{i \in I}$ existing in the artificial networks described in Section 4.2, joining them should cause a decrease of modularity, hence

$$\Delta Q^{w}_{c_i^1,c_i^2} \leq 0 \qquad \text{for } i \in I \tag{2.80}$$

where I is the parameter defining number of pairs of small communities to be selected from the artificial network. Using the penalty method, the optimization problem can be formulated as

$$\min_{w} \quad F(w) = (\overline{w} - 1)^2 + \lambda_1 \sigma_w^2 + \lambda_2 \sum_{1 \leq i \leq I} h(\Delta Q_{c_i^1,c_i^2}) \tag{2.81}$$

where $w = \{w_e\}$ is the set of the weights of edges in the entire graph, σ_w^2 is the variance of the edge weights, $\overline{w}$ is the average edge weight $\frac{\sum_{e \in E} w_e}{|E|}$, $h(x)$ is the loss function such as the sigmoid function $h(x) = \frac{1}{1+e^{-x}}$, λ_1 is a constant, and λ_2 is a coefficient for the penalty terms. The regularization term $(\overline{w} - 1)^2$ ensures that the resulting average edge weight is close to 1. Using regularization on $\overline{w}$ directly is likely to result in very small weights that are inconvenient in community detection tasks. For the same reason, the regularization on the variance of edge weights σ_w^2 limits the total number of negative edges. When $\lambda_1 \gg \lambda_2$, the above optimization problem converges at $w_e = 1$ for $\forall e \in E$, yielding the weights of edges in an unweighted graph. So far, we have presented an optimization method of modifying edge weights which helps avoiding improper merging of communities. However, it involves as many variables as the total number of edges and does not guarantee that edges with similar features have similar weights. Let's denote the i-th topological feature of one edge e as $x_e^{<i>}$.

The weight of an edge e is obtained by the linear regression

$$w_e = p_0 + \sum_{i=1}^{r} p_i x_e^{<i>} \tag{2.82}$$

where p_i is the parameter of the i-th feature, for $i = 1, \ldots, r$. Let the feature vector of an edge e be $x_e = (1, x_e^{<1>}, x_e^{<2>}, \ldots, x_e^{<r>})^T$. Then Eq. 2.82 can be rewritten as

$$w_e = p^T x_e \tag{2.83}$$

where vector $p = (p_0, p_1, \ldots, p_r)^T$. This way, the objective function in Eq. 2.81 becomes a function over p. The first order partial derivative of the objective function over p_i is

$$\frac{\partial F(w(p))}{\partial p_i} = \frac{\partial F(w)}{\partial w} \times \frac{\partial w}{\partial p_i} \tag{2.84}$$

The second term on the right side of the above equation is

$$\frac{\partial w}{\partial p_i} = \left(x_1^{<i>}, x_2^{<i>}, \ldots, x_{|E|}^{<i>} \right) \tag{2.85}$$

The first term on the right side of Eq. 2.84 is

$$\frac{\partial F(w)}{\partial w} = \frac{\partial (\overline{w} - 1)^2}{\partial w} + \lambda_1 \frac{\partial \sigma_w^2}{\partial w} + \lambda_2 \sum_{i \in I} \frac{\partial h(\Delta Q^w_{c_i^1, c_i^2})}{\partial \Delta Q^w_{c_i^1, c_i^2}} \frac{\partial \Delta Q^w_{c_i^1, c_i^2}}{\partial w} \tag{2.86}$$

where the partial derivative $\frac{\partial h(\Delta Q_{c_i^1, c_i^2})}{\partial \Delta Q_{c_i^1, c_i^2}}$ is obtained from the specific loss function $h()$. It is also easy to compute the partial derivative $\frac{\partial \Delta Q^w_{c_i^1, c_i^2}}{\partial w}$ according to Eq. 2.79. **Algorithm**. To solve the optimization problem presented above, we can apply a quasi-Newton method, such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [63], which requires only the first derivative of the objective function to find the optimal result. The pseudo code of the training algorithm is presented in Algorithm 0. During the training phase, $|I|$ pairs of ground truth communities $\{c_i^1, c_i^2\}_{i \in I}$ can be chosen randomly from the artificial network assuming that the ground truth communities are provided. One efficient way to obtain the required number of pairs of ground truth communities is to sample adjacent communities in the artificial network randomly until $|I|$ pairs are collected. As indicated by Theorem 3, small

communities are preferred to large communities here. Hence, we can set an upper bound on the size of the chosen communities. After the parameters are inferred, the regression model which assigns weights to edges can be applied to enhance the performance of community detection algorithms.

Algorithm 1 Regression Model Training Algorithm

```
Initialize p
for each edge e do
    x_e ← extracted features of edge e
    w_e ← p^T x_e
end for
tol ← 0.0001
Construct {c_i^1, c_i^2}_{i∈I}
repeat
    Compute ∂F/∂w using Eq. 2.86
    ∂F/∂p ← ∂F/∂w × ∂w/∂p
    Update p via one BFGS step
    for each edge e do
        w_e ← p^T x_e
    end for
until ||∂F(w(p))/∂p|| < tol or the maximum number of iterations is made
```

The **time complexity** of Algorithm 1 is $O(k(|I| + |E_a|))$ where k is the number of BFGS iterations before the algorithm converges, $|I|$ is the number of constraints in Eq. 2.80 and $|E_a|$ is the total number of edges in the artificial graph. In order to accelerate the computation, we adopt the following key speedup improvements. To compute the change of modularity upon joining two communities, the weights of all edges in the artificial graph need to be summed up which takes significant amount of time in each BFGS step. The summation of weights is

$$W = \sum_{e \in E_a} w_e = \sum_{e \in E_a} p^T x_e = p^T \sum_{e \in E_a} x_e \tag{2.87}$$

which can be calculated efficiently because $\sum_e x_e$ needs to be computed only once at the outset of the optimization process, and W is re-computed as the inner product of p and $\sum_e x_e$ in every iteration. The sums of weights of the edges related to each community c, such as W_c^{in} and W_c^{out}, and the variance of edge weights σ_w^2 can also be computed in the similar manner. Note that such speedup can be achieved because we intentionally use linear

Table 2.3. Summary of the networks.

No.	Network	#Nodes	#Edges	Type
1	American college football	115	613	Real
2	LFR benchmark	5000	≈35000	Synthetic
3	Amazon product co-purchasing network	334863	925872	Real
4	DBLP collaboration network	317080	1049866	Real

regression to compute the edge weights in Eq. 2.82. Otherwise, if non-linear regression function is used to obtain the edge weights, Eq. 2.87 does not hold and it generally takes more time to obtain the sums of weights. In our algorithm, the edges with both endpoints not in any communities in pairs $\{c_i^1, c_i^2\}$ for $i \in I$ are not involved in the computation of every BFGS iteration. The number of edges involved in every BFGS iteration is at most $2|I|Z$ where Z is the average number of edges in communities in pairs $\{c_i^1, c_i^2\}$. So, the time complexity is reduced to $O(|E_a| + k|I| \times 2|I|Z) = O(|E_a| + k|I|^2 Z)$. In practice, this accelerated algorithm provided at least a 50-fold speedup compared to Algorithm 1.

The state-of-art greedy modularity maximization algorithm, Fast Greedy [5], is executed on several real and synthetic networks. The details of the tested networks are summarized in Table 2.3. We evaluate the execution time of the training of the regression model and the additional time needed to convert an unweighted graph into a weighted one. We does not report the time cost of community detection, which depends on the specific algorithms. Hence, the reported time cost consists of two parts: (i) Training: the time cost to infer all the parameters of the regression model from the artificial graph; (ii) Weighting: the time cost to compute the weights of every edge in the original graph. Note that both parts include the I/O cost of loading the network files from disk and the edge topological feature extraction time.

The LFR benchmark networks [44] serve as one of the standards for the evaluation of community detection algorithms. The properties of the network generated from the benchmark are defined by the following three parameters: γ which is an exponent of the node degree in the power law distribution, β which is an exponent of the community size in the power law distribution, and μ which is the mixing parameter that defines the fraction of edges originating in a community that have one endpoint outside of it. In our experiments, every LFR benchmark network has 5,000 nodes with the average node degree 15 and the maximum node degree 50. The exponents γ and β are set as 2 and 1 respectively and the

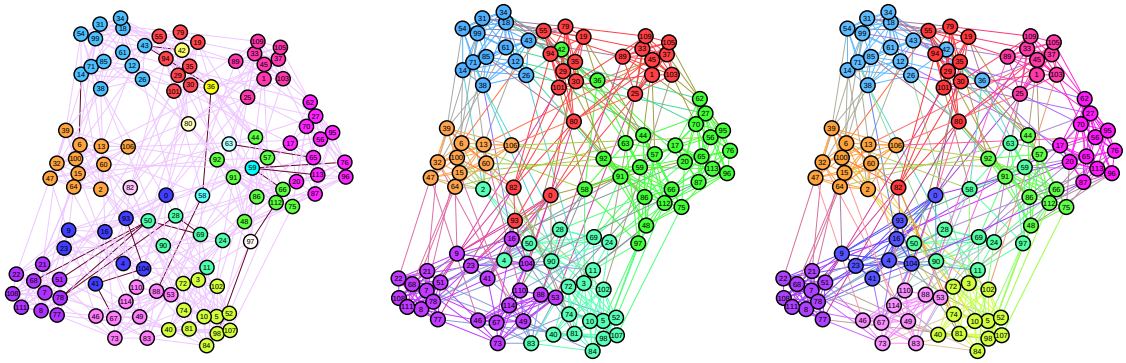
mixing parameter μ takes two values, 0.45 and 0.5, which are quite challenging because high values of the mixing parameter are likely to result in loose community structures. Considering the randomness in the generation of synthetic networks, 10 network instances are constructed for each μ value.

We evaluate the modularity maximization performance on the original unweighted LFR benchmark networks. Table 2.4 lists the results of the following algorithms Fast Greedy modularity maximization algorithm (FG) on the original unweighted graphs and Fast Greedy modularity maximization algorithm (FG-w) running on the weighted graphs produced by our model. As seen in Table 2.4, the performance of the Fast Greedy (FG) algorithm [5] has been significantly improved by maximizing the modularity on the weighted networks instead of on the original unweighted graph. The F-measure is improved by nearly 40% and the NMI metric is improved by 25% in all cases. In Table 2.4, the modularity Q and modularity density Q_{ds} values are all computed over the original unweighted LFR benchmark networks. This surprising result shows that the execution of Fast Greedy algorithm on weighted graph can improves the Q_{ds} value for the corresponding unweighted graph. In other words, the weighted edges allows the greedy algorithm to escape from local maximum of Q_{ds} and get better value of it on the original unweighted graph. Also, using the edge weighting scheme, the Fast Greedy algorithm which maximizes the weighted modularity performs better than the fine-tuned Q_{ds} algorithm [6]. In addition, we compared our approach to the previously published algorithm CNM [61]. The introduced here edge weighting scheme achieves an average 85% Jaccard-index score while the CNM algorithm obtains the average score of 82% on 10 different realizations of the LFR benchmark networks using the same parameters, with the mixing parameter set to $\mu = 0.5$. The remaining specific construction parameters of these LFR benchmark networks and the definition of Jaccard-index can be found in [61].

American college football network The American college football network [45] consists of 115 nodes representing college football teams playing in a league with 11 conferences. Two teams are linked if they have played with each other in the year 2000 season. The teams in each of the 11 conferences can be treated as one community because they play with each other often. There are 8 independent teams (not members of any conference), each forming a single community. 19 ground truth communities are shown in Figure 2.13.a with each color representing a single community. However, only 6 communities are detected by the Fast Greedy algorithm on an unweighted graph as shown in Figure 2.13.b because some

Table 2.4. Metric values characterizing the community structures computed over the original unweighted LFR benchmark networks but discovered by different algorithms.

μ	Method	VI	NMI	F-measure	ARI	Q	Q_{ds}
0.45	FG	3.2135	0.5953	0.3379	0.2355	0.4214	0.0366
	Fine-Tuned Q_{ds}	1.1523	0.8925	0.8806	0.7337	0.4536	0.1632
	FG-w	0.0137	0.9987	0.9990	0.9972	0.5152	0.1668
0.5	FG	3.5187	0.5481	0.2937	0.1993	0.3739	0.0274
	Fine-Tuned Q_{ds}	1.9677	0.8036	0.7489	0.4984	0.3563	0.1196
	FG-w	0.0678	0.9934	0.9950	0.9864	0.4625	0.1381



(a) **(b)** **(c)**

Fig. 2.13. The communities detected by the Fast Greedy algorithm [5] in the American college football network [45]. Nodes are colored according to communities to which they have been assigned. (a) 19 ground truth communities defined as 11 conferences and 8 independent teams. Edges in black are assigned negative weights by the edge weighting scheme. (b) 6 communities detected on the unweighted graph by the modularity maximization method. (c) 11 communities detected on the weighted graph by the modularity maximization method.

adjacent ground truth communities are joined together. The regression model converts the original unweighted graph to a weighted graph where the edges with negative weights are marked in black in Figure 2.13.a. On the weighted graph, the Fast Greedy algorithm can find 11 league communities, each containing one individual conference, although it allocates the independent teams to some of these league communities. The regression model is trained by sampling the ground truth communities on the artificial SBM network, which is constructed

Table 2.5. Metric values characterizing the community structures computed over the original unweighted American college football network but discovered in either the original unweighted graph or the corresponding weighted graph produced by our model.

Metric	Graph	FG	LE	LP	RW	ML
NMI	Original	0.58528	0.58140	0.76962	0.83833	0.83391
	Weighted	0.91117	0.85903	0.92635	0.91117	0.87272
ARI	Original	0.49333	0.49441	0.71749	0.86938	0.85815
	Weighted	0.94723	0.88982	0.91539	0.94723	0.90085
Q	Original	0.56860	0.49326	0.57668	0.60337	0.60503
	Weighted	0.60140	0.59338	0.57315	0.60140	0.60356
Q_{ds}	Original	0.15877	0.13661	0.21106	0.23650	0.23626
	Weighted	0.25696	0.23893	0.24025	0.25696	0.24889

to be similar to the Football network. The training process takes approximately 10 seconds on a machine with a single 2.5GHz CPU. Table 2.5 lists the results of the followings approaches: Fast Greedy algorithm [5] (FG), leading eigenvector method [56] (LE), label propagation algorithm [64] (LP), community detection based on random walks [65] (RW), multilevel algorithm [51] (LW) measured by the normalized mutual information (NMI) and adjusted rand index (ARI). As illustrated in Table 2.5, in addition to the Fast Greedy modularity maximization algorithm, the state-of-the-art community detection algorithms, including label propagation algorithm by Raghavan et al. [64], Newman's leading eigenvector method [56], the algorithm based on random walks [65] and the multilevel algorithm by Blondel et al. [51], also demonstrate improved performance on weighted graphs produced by our method. This result additionally supports our claim that properly weighting a graph can lead to an improved quality of community detection. Note that, in the experiments, the edges with negative weight are removed from the graph for community detection algorithms which are not able to handle negative weights due to the algorithm design or implementation.

For a fair comparison, regardless of whether the partition of the graph is determined with or without the edge weights, both the modularity Q and modularity density Q_{ds} are computed over the unweighted graph, i.e., edge weights are all set to 1. Hence, a better Q or Q_{ds} found on the weighted graph indicates the edge weights allow the maximization algorithm to avoid the inferior local optima. The NMI and ARI measures indicate that the communities detected in the weighted graph are generally accurate. However, from the aspect of modularity, for three algorithm, LP, RW and ML, such communities may be evaluated

as inferior (as they have slightly lower modularity) than the communities discovered in the original unweighted graph. Consequently, even if the maximum modularity is reached in the original unweighted graph, the resulting communities are still not likely to match the ground truth. In contrast, the modularity density Q_{ds} of the communities detected in the weighted one is higher than in the original unweighted graphs, which means that it accurately measures the quality of these communities. The proposed edge weighting scheme leads to a higher modularity density Q_{ds} in all cases because the weighted edges allows the greedy algorithm to escape from local maximum of Q_{ds} and get better value of it on the original unweighted graph. **Large Networks** We evaluate the performance of our model on two

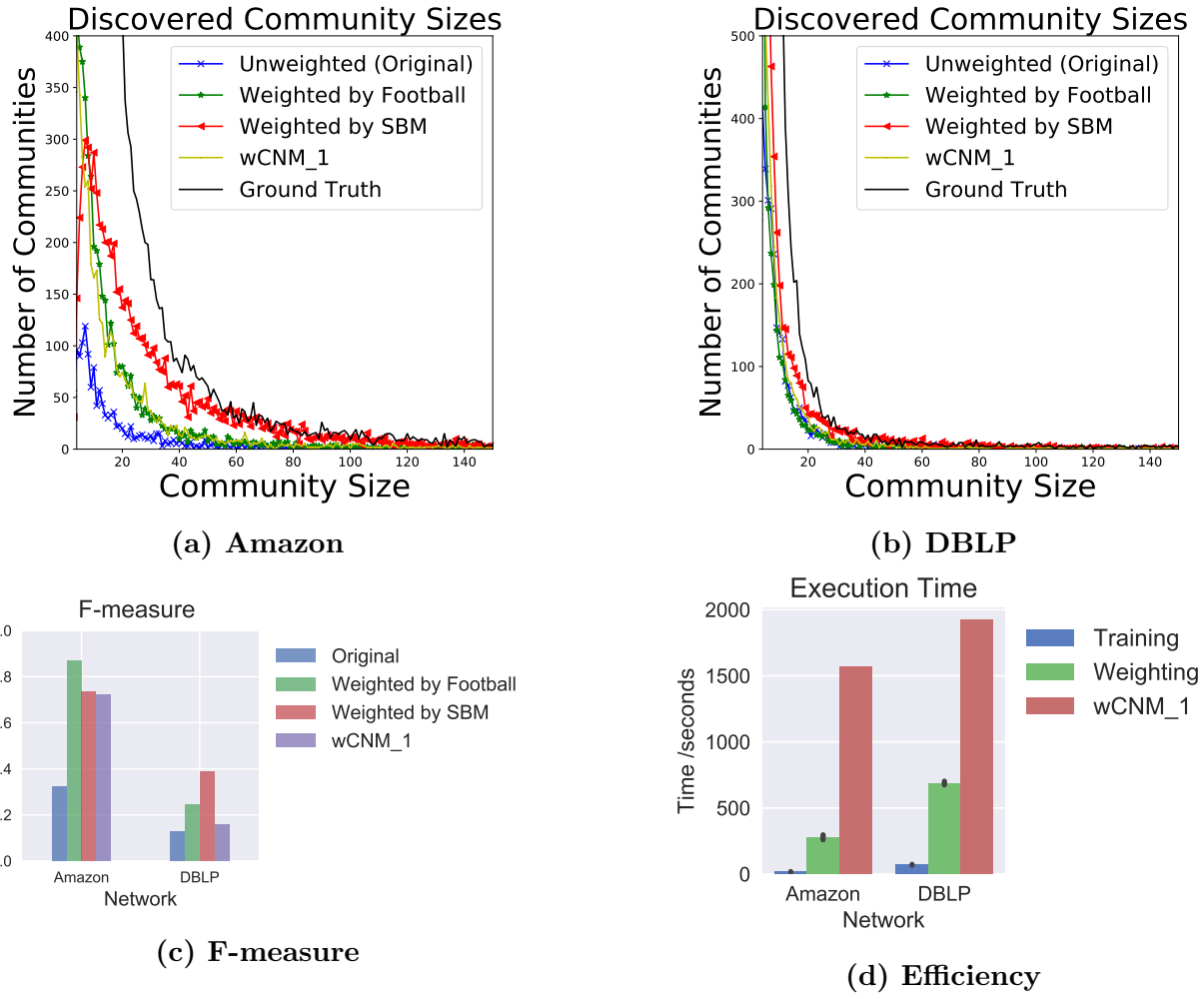


Fig. 2.14. Performance improvement of community detection in Amazon and DBLP networks.

large real networks: Amazon co-purchasing network and DBLP co-authorship network. The Amazon co-purchasing network [66] consists of 334,863 products with two frequently co-

purchased products linked by an undirected edge. Each collection of products from the same category forms one ground-truth community. The DBLP collaboration network [66] is the co-authorship network where every node represents a researcher. Two researchers who published at least one paper together are linked. Following others, we assume that individual ground-truth community is defined by the publication venue, e.g., journal or conference. The proposed weighting scheme is compared with the wCNM_1 algorithm [61] which computes the weight of an edge using all the triangles and 4-cycles containing it. In our experiments, the wCNM_1 algorithm iterates only once over updates, because the results in [61] show that additional iterations negligibly improve the final results. The regression model which converts the original unweighted graphs to weighted ones is trained by sampling the dynamically constructed artificial SBM networks. In addition, we also test the performance of our model trained by the ground truth communities in the American college football network, as shown in Figure 2.14. Perhaps surprisingly, the accuracy of the modularity maximization algorithm on the weighted graph when weights were based on SBM artificial network has improved for the Amazon network by almost 50% as measured by F-score and even more for the DBLP network. The sizes of communities discovered in Amazon and DBLP networks containing more than 3 nodes are plotted in Figure 2.14a-b. In the weighted graph produced by our model for the Amazon network, the distribution of the sizes of the detected communities is close to the distribution of the sizes of ground truth communities for weights based on SBM artificial network but quite different for weights based on the Football network. Since the F-score was similar for those two cases, this result demonstrates the importance of inspecting the distribution of the community sizes. In case of the DBLP network, the improvement of F-score is significant for the weights based on the SBM network, but the distribution of the community sizes is different. We believe that these two results show that presumed ground truth communities in DBLP are not correct, and that smaller communities of researchers co-authoring papers across several venues are the right communities. These results show that our model successfully converts large networks to the weighted ones where the modularity maximization algorithms can perform better than they do on the original unweighted networks. In general, these large networks can be processed in a few minutes as shown in Figure 2.14d. The computation time is divided into two parts: (i) Training: the time spent to infer all the parameters of the regression model; (ii) Weighting: the time needed to compute the weights of every edge in the graph. Both steps include the I/O processing

time of loading the network files from disk. The edge topological feature extraction (i.e., weighting) time increases as the number of edges grow, therefore processing of dense graphs can be more time-consuming. Unlike the weighting time, the training time does not change much with the size of the original network. This is because the size of the constructed artificial network is independent of the size of the original input graph. Last but not least, in our experiments, the edge topological feature extraction and edge weight evaluation use a single thread implementation. However, as problems that are easily parallelizable, they can be partitioned into many individual tasks to achieve a better performance.

2.7 Conclusions

Modularity maximization is one of the state-of-the-art approaches for community detection in complex networks. However, due to the resolution limit problem, it tends to merge small, well-formed communities into a large component to increase the modularity. To deal with the resolution limit problem, we propose two key enhancements to the modularity maximization - one assigning proper weights to edges in the network; the other recursively dividing a graph into subgraphs until the remaining communities are statistically significant. The experimental results show that both approaches significantly improves the quality of the detected communities in the real and synthetic networks.

Besides the contributions on the practical community detection algorithms, we also develop the theoretical results for modularity-based community detection. Specifically, we establish the asymptotic lower and upper bounds of the generalized modularity. The lower bound corresponds to the highest background inter-community edge density in a network, while the upper bound corresponds to the lowest intra-community edge density. This work connects the resolution limit of modularity with random graph theories, which also reveals the "plateaus" problem that no universal resolution parameter exists to detect all centers. The issue is analogous to finding mountains that are located at different plateaus; using a single altitude either would miss the lower mountains, or would treat the higher peaks as one mountain.

The stochastic block model can produce a wide variety of different network structures, including traditional assortative communities and different from them disassortative structures. In theory, it should be possible to guide the inference algorithm which type of structure is preferred when both types of structures fit the stochastic block model and

its degree-corrected variant well for the input network. However, the existence of multiple local optima of the log-likelihood traps the inference algorithms in one of them. We apply a simple yet effective constraint on nodes' internal degree ratio of the degree-corrected stochastic block model. The resulting algorithm reliably finds assortative or disassortative structure as directed by the value of a single regularization parameter. We validated the model experimentally testing its performance on several real and synthetic networks.

Future works include applying more capable models as the nested hypothesis in the significance testing framework and extending the multi-scale community detection algorithm and edge weighting scheme to directed networks.

CHAPTER 3
INFORMATION PROPAGATION AND VIRALITY PREDICTION

3.1 Introduction

With the rise of online medium sites such as Facebook and Twitter, information propagation in these platforms attracts the interests of many researchers. Understanding the information propagation in these systems does not only provide better insights into the underlying sociology but is also crucial for the prediction of emergent social, economic and political phenomena. In the previous section, we study the detection of community structures in the complex network. In contrast, here, based on the topological features, we seek to predict the virality of cascades over the information propagation networks.

The mutual excitations between network nodes are widely observed in a variety of networks, ranging from the diffusion of purchasing behavior among friends to the spread of contagious disease. In a social network, information propagation can also be explained by the mutual excitations between individuals. Many previous works [67]–[69] model the information diffusion as epidemics on networks - the acceptance of information is viewed as an infection of a node by infected neighbors in the network. These works assume that there exists an explicit propagation pathway which is sufficient to explain the observed information diffusion [66] - messages can only spread along the predefined edges between nodes. In this section, we adopt these assumptions and develop efficient algorithms to predict the viral information propagation at its early stage.

As shown in [70], strong community structures in a network enhance the local, intra-community spreading. This phenomenon is supported by the observations in online media: most news events are reported by the online news sites in the same region, using the same language. We exploit such community structures to parallelize the derivation of node em-

Portions of this chapter previously appeared as:

X. Lu and B. K. Szymanski, "Towards limited scale-free topology with dynamic peer participation," *Comput. Netw.*, vol. 106, pp. 109121, 2016.

X. Lu and B. K. Szymanski, "Predicting viral news events in online media," *Parallel and Distrib. Process. Symp. Workshops (IPDPSW)*, pp. 14471456, 2017.

X. Lu and B. K. Szymanski, "Scalable prediction of global online media news virality," *IEEE Trans. Computat. Social Sys.*, no. 99, pp. 113, 2018.

beddings because most cascades occur in local communities. In Section 3.2, according to the frequency of co-occurrence in cascades, the nodes in a network are divided into multiple communities in which the propagation occurs frequently. Every node in the propagation network has two latent vectors, one representing its influence to other nodes on different topics and the other representing its selectivity upon the inputs from other nodes on different subjects [71]. Such latent representations of the influence and selectivity are analogous to the node's vector representation in the non-negative matrix factorization approach [72]–[74], where the inner product of two vectors indicates the strength of the connection between the corresponding nodes. These node embeddings are then used as input to predict the size of cascades. Unlike [75], we represent the possible parameters of one link by the influence and selectivity of its two endpoints, which reduces the number of possible parameters and permits an efficient parallel algorithm to infer them.

In Section 3.2, we propose a scalable community-based probabilistic framework to model the spreading of news about events in online media. News reports shape the public perception of the critical social, political and economic events around the world. The way in which emergent phenomena are reported in the news makes the early prediction of such phenomena a challenging task. Our approach exploits the latent community structure in the global news media and uses the affiliation of the early adopters with a variety of communities to identify the events widely reported in the news at the early stage of their spread. The time complexity of our approach is linear in the number of news reports. It is also amenable to efficient parallelization. To demonstrate these features, the inference algorithm is parallelized for message passing paradigm and scales well on RPI Advanced Multiprocessing Optimized System (AMOS), one of the fastest Blue Gene/Q supercomputers in the world. Thanks to the community-level features of the early adopters, the model gains an improvement of 20% in the early detection of the most massively reported events compared to the feature-based machine learning algorithm. Its parallelization scheme achieves orders of magnitude speedup.

The scale-free topology has some good properties to facilitate information propagation, including high tolerance to random attacks [76], high synchronizability [77] and resistance to congestion [78]. For this reason, several growth models are proposed to construct the scale-free overlay topology. The BA model [79] manages to explain the evolution of scale-free topologies by a core principle named "Preferential Attachment" which means a new

node is more likely to connect to heavily linked nodes when it joins the network. However, it is not practical in real distributed applications because the global information is required to maintain the degree distribution. Therefore, a distributed approach that constructs the topology without global information is desired.

In Section 3.3, we propose a distributed algorithm to construct limited scale-free networks which are robust against the traffic congestion and network failures [15]. We compare our approach with other P2P network construction algorithms including HAPA [80], Gaian [81] and SRA [82] algorithms which construct the scale-free overlay topology with partial or no global information. The results show that our approach can maintain a scale-free overlay network while the other approaches are often vulnerable to dynamical peer participation.

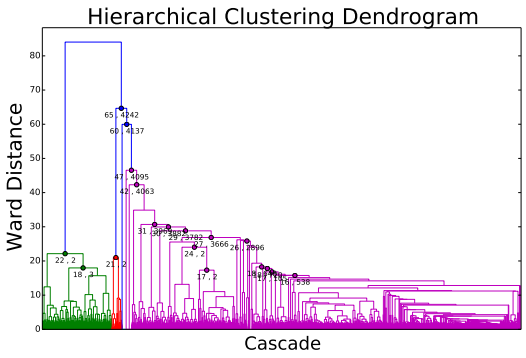
3.2 Scalable prediction of global online news

3.2.1 Datasets

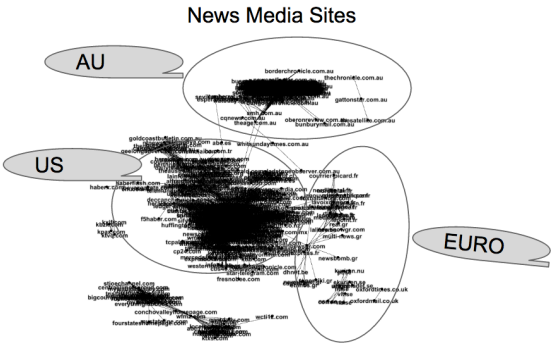
We investigate the news data in the Global Database of Events, Language, and Tone (GDELT) project [83]. The GDELT project provides real-time news events extracted from tens of thousands of online news sites around the globe. Advances in natural language processing allows the sentiment analysis, real-time translation of 65 languages and the identification in events data of named entities including organization, location, count and theme. The dataset is currently available on Google Cloud Platform. Users can use Google BigQuery to download specific tables or process the data remotely by SQL commands. The GDELT database contains records the mentions of news events by news sites since February 19, 2015 to present. Here we present some unique features explored in these records.

- **Emergence of news events**. We calculate the duration of events, computed as the period since its earliest record until the latest. Most news events are reported by the news site within the first 50 hours. This reflects the short life cycle of the production and consumption of the news events. One explanation for this phenomenon is that a news site would prefer not to report an event which is considered out-of-date. In contract to the breaking news, the discussion of long-term topics such as greenhouse effect lasts a very long time, but these events constitute a very small portion of the dataset.

- **Most cascades are local**. For a period covering one year, we randomly choose 5,000



(a) Dendrogram of hierarchical clustering of the 5,000 random cascades of news events



(b) Backbone network of the news sites associated with the 5,000 sampled cascades of news events

Fig. 3.1. The hierarchical structure of online media and the backbone co-reporting network.

news events in the GDELT dataset. Any two news sites reporting at least 50 events together are linked in the graph shown in Figure 3.1b. Clearly, there are four clusters visible in Figure 3.1a, one standing for the news sites in the U.S., one for the news sites in Australia and the other for the news sites in European countries, while the remaining one is a mixture of sites in different regions. Initially, the pair-wise distance between any two cascades of events is measured by the Jaccard-index

$$\text{Jaccard} = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} \tag{3.1}$$

where $N(j)$ is the set of news sites reporting the news event j. Then the hierarchical clustering algorithm [84], which merges iteratively the closest cascades according to the Ward distance measure among all pairs of cascades, is applied to obtain a dendrogram as shown in Figure 3.1a. The Ward distance and the number of cascaded contained in a cluster are also plotted in text at some associated inner nodes of the clustering tree. There are three obvious clusters shown in the dendrogram. The largest cluster in blue corresponds to the news sites in the U.S., while most news sites in Australia are located in the middle cluster in green. The news sites in U.K. and European countries are placed in the left-most cluster in red. Although the plot shows only a raw distribution of the news site in different clusters, the dual visualization indicates that the hierarchical structures of cascades correlate with the community structure of

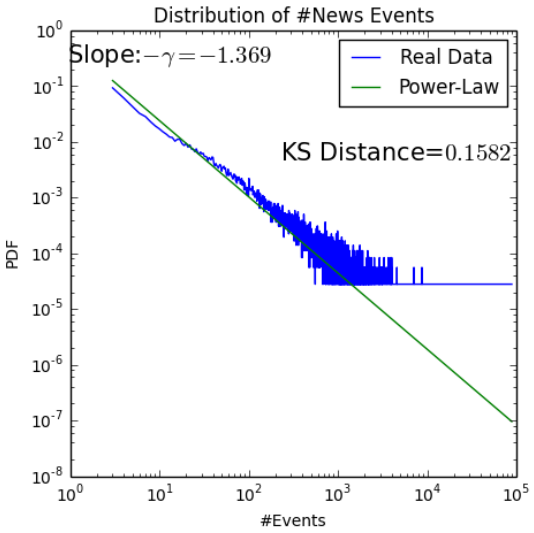


Fig. 3.2. The distribution of the number of events reported by the news sites follows the power law.

co-reporting networks of the news sites. That is, as an illustrative example here, most cascades of news events occur frequently within certain regions only.

- **Matthew effect in online media**. The term Matthew effect (or accumulated advantage) originated in sociology and refers to a phenomenon in which "the rich get richer and the poor get poorer." In network science, it refers to preferential attachment of earlier nodes in a network [79] where nodes that few nodes that acquire more connections than others will increase its connectivity at a higher rate while others have few connections. As seen in Figure 3.2, the distribution of the number events reported per site follows the Power Law, which has been widely observed in a variety of networks. As shown in Figure 3.2, only a few popular news sites have reported millions of events, while most of the news site reported 5,000-10,000 events. In the Figure 3.2, the news site which reported less than 5,000 events in one year are ignored because they are less influential than the remaining ones. The goodness-of-fit is measured by the p-value as shown in Figure 3.3.

We also generate **synthetic information cascades** to evaluate the performance of the prediction algorithms proposed in this section. The networks are generated using the Stochastic Block Models as discussed in Section 2.2.2. Given the membership of all the n nodes, the edge with two endpoints in different communities is created with the probability

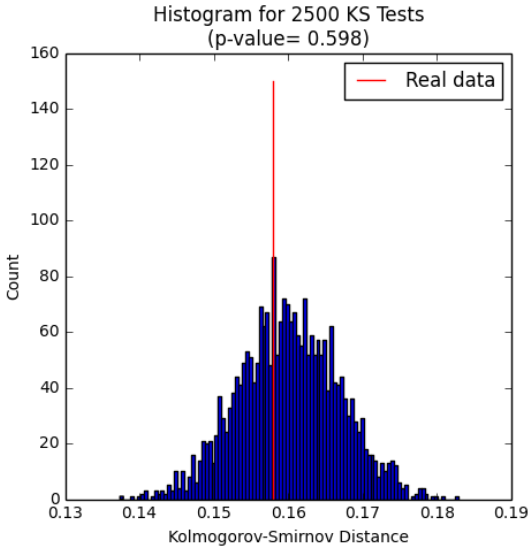


Fig. 3.3. Goodness of fit measured by the p-value.

β and the edge inside a community is created with probability α. In most cases, α is much larger than β, so the graph contains the dense intra-community structures and sparse inter-communities structures. We create the SBM graphs containing 2,000 nodes with $\alpha = 0.2$ and $\beta = 0.001$. The average degree of nodes is approximately 10. Each community in SBM graph contains approximately 40 nodes.

On each network instance, the spreading process is simulated according to the stochastic propagation model proposed by Kempe et al. [85]. Since any cascade would eventually flood the entire network, we set an observation window during for the cascades. After the observation window, the current spreading process will be terminated instantly and a random node is chosen as the initiator to start the simulation of the next cascade. A total of 3,000 cascades are collected for each graph instance. The first 2,000 cascades are used to infer the influence and selectivity vectors of nodes in the network and the last 1,000 cascades are used to test the accuracy of the virality prediction algorithm. For the last 1,000 cascades, the infections occurring in the first 2/7 time of the observation window are provided for the prediction task, while the remaining parts of infections are assumed to be unknown.

More formally, the infection delay of v, i.e. t_v, can be treated as a random value

$$\begin{aligned} t_v &= \min t_{u_1}, t_{u_1}, \ldots, t_{u_r} \\ \forall i \in [1, r] \quad &: \quad t_{u_i} \sim \mathcal{K}(\alpha_{u_i,v}) \end{aligned} \tag{3.2}$$

where t_{u_i} is the infection time of the i-th neighbors of node v, $\alpha_{u_i,v}$ is a parameter associated with the edge (u, v) in the propagation network and $\mathcal{K}()$ is the distribution of infection delays. In our experiments, $\mathcal{K}()$ is set as the exponential distribution which is observed in many social dynamics [86], and we set $\forall(u, v) \in E$: $\alpha_{u,v} = 1$ for simplicity. In theory, the IC model supposes the entire network will be infected given a sufficiently long period. Since contagions generally do not infect all the nodes in the network, in our experiments the simulation of every cascade happens within a predefined observation window [75].

3.2.2 Community affiliation model for information cascades

Community structures are widely observed in a variety of networks. Based on patterns of news about events spreading in the online news media, we present two basic observations below:

- Most news sites have a preference for the topics of their news coverage, e.g. politics, finance, education, military, technology or sports.
- The news reports are usually confined to the geographical and cultural boundaries. Although many media companies have ambitions to have global market presence, most media sites have only a regional reach [87].

Much like many community detection methodologies [5], [7], [46], [51], [88] which aim at discovering dense sub-graphs embedded in a network, we seek to recover such community structure for the online news sites. However, in the global and local news markets [83], connecting every two sites reporting the same event could result in an extremely dense network. Finding community structure in such network would not only incur high computational power but would also cloud the real connections among communities. Therefore, we find the latent community affiliation of these news sites rather than drawing the edges between specific news sites. Since the underlying network topology is unknown, our probabilistic model incorporates the time points of each infection in the observed news cascades, i.e., the time of each news report. Formally, we present the definition of information cascade below.

An information/news cascade is a set of infections $\{(u, t_u)\}$. Each infection (u, t_u) consists of a node u and its infection time t_u. Our model assumes the news cascades happen at the community level. For a particular community of news sites, the probability of a site reporting a particular news depends on both the affiliation of this news sites with the

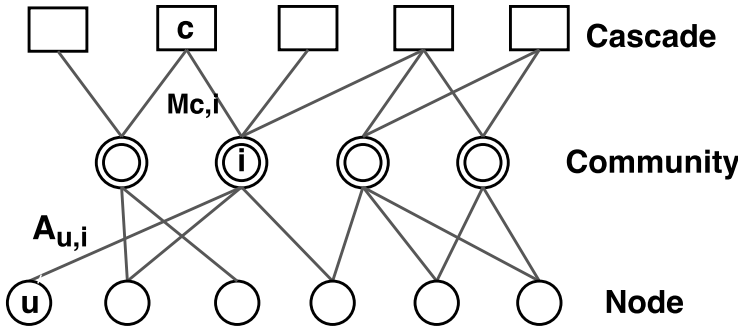


Fig. 3.4. Community affiliation model with cascades. Cascade c belongs to the two leftmost communities, and node u belongs to community i.

community and the probability of the news cascade reaching its community. To formalize this idea, we define latent community below.

A latent community is a set of nodes that, conditioned on the infection of one member, the probability that other members will get infected within a limited period is much higher than for the non-members. In [74], authors propose a community affiliation model where each node has a probability belonging to one of the overlapping communities in the network. We extend this community affiliation model by taking into account the probability for each cascade to spread to these overlapping latent communities. As illustrated in Figure 3.4, if a cascade has a non-zero probability of spreading to a community, this cascade (square) is connected to the community (double circle); if a node (circle) belongs to a community, then this node is connected to the community as well. Our affiliation model captures two important features of cascades and communities in complex networks: (i) the community structures are highly overlapped, a node can belong to multiple communities at the same time; (ii) the members of the same community tend to accept similar information during the information cascades.

Social theories [86] suggest that the response time of human beings usually follows the exponential distribution, which explains the burstiness of social behavior in many scenarios. Our analysis of 723,037 randomly sampled news reports shows that the delays of news reports also fit the exponential distribution with a high R^2 score of 92%. We consider the news reports with a delay less than 21 hours, which comprise 99% of the sampled data from the Global Database of Events, Language, and Tone (GDELT) dataset. Based on this observation, we model the response time of news sites to events by an exponential distribution.

For every single community i, let $A_{ui} > 0$ denote the strength of affiliation of node u with it and $M_{ci} \in (0, 1]$ denote the probability of the cascade spreading to community i. In this community, the response time of node u to cascade c follows the exponential distribution

$$t_u^c(i) \sim Exp(A_{u,i} M_{c,i}) \tag{3.3}$$

which indicates that the node u gets quickly infected when the cascade c is highly likely to reach a community i, i.e. $M_{c,i}$ is large, and the affiliation of u with community i is strong, i.e. $A_{u,i}$ is large.

Most real-world networks have overlapping community structures which allow a node to belong to many communities. Therefore, in our model, the response time of a node to a cascade corresponds to the minimum response time in all communities. Given a node u, a cascade c and a total of m overlapping communities, the minimum response time also follows the exponential distribution. This is because the minimum of m mutually independent random variables $X_i \sim Exp(\lambda_i)$ for $i = 1, 2, \ldots, m$ also follows the exponential distribution, i.e. $min(X_i) \sim Exp(\sum_i \lambda_i)$.

$$\min\{t_u^c(1), t_u^c(2), \ldots, t_u^c(m)\} \sim Exp(\sum_{i=1}^{m} A_{u,i} M_{c,i}) \tag{3.4}$$

where the rate parameter is the sum of the rate parameters for each individual exponential distribution in Eq. 3.3.

In reality, news agencies usually have some bias for the content of information they spread. Such bias can be positive - there are certain types of information a news agency favors, while it can also be negative - messages of no interest to agency's audience are ignored or intentionally blocked. It is tempting to allow the value of $M_{c,i}$ to be negative for this reason so that the affiliation with some community may delay the response time of some node, i.e. $A_{c,i} M_{c,i} < 0$. However, it could result in a negative rate parameter λ in Eq. 3.4 which violates the constraint $\lambda > 0$ of the exponential distribution. Hence, we smooth the rate parameter via the sigmoid function $\sigma : \mathbb{R} \to [0, 1]$. In this way, given an information cascade c, the response time of a node u, $t_u^c = \min t_u^c(i)$, draws from an exponential distribution with a rate parameter $\gamma = w\sigma(A_u \cdot M_c)$,

$$t_u^c \sim Exp(w\sigma(A_u \cdot M_c)) \tag{3.5}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, w is a scaling parameter and $\cdot$ represents the inner product in vector space, while the i-th component of vector A_u and M_c are A_{ui} and M_{ci} respectively. Using the sigmoid function here avoids any constraint for the parameters A_u and M_c, and most importantly, it improves the robustness of the parameter estimation because the sigmoid function is differentiable at each point.

So far, the model takes into account the participants of a cascade. However, the nodes which are not involved in a cascade also provide information about their affiliations with different communities. Since these silent nodes are equivalently important, we set a upper bound on the response time, $T \gg t_u^c$ for $\forall u$ and $\forall c$, such that all the silent nodes during the cascading propagation can be assumed to have this long response time T.

Given an information cascade $c = \{(u_i^c, t_i^c) | i = 1, 2, \ldots, n_c\}$ where the i-th node u_i^c has response time t_i^c, the likelihood of observing a cascade c is

$$L_c = \prod_{u \in V_c} p_{u,c}(t_u^c) \prod_{u \notin V_c} p_{u,c}(T) \tag{3.6}$$

where $V_c = \{u_i^c | i = 1, 2, \ldots, n_c\}$ denotes the set of nodes involved in cascade c and $p_{u,c}(t)$ is the probability density function of the exponential distribution

$$p_{u,c}(t) = w\sigma(A_u \cdot M_c) e^{-w\sigma(A_u \cdot M_c)t} \tag{3.7}$$

Since the likelihood L_c is the product of $|V|$ terms, Eq. 3.6 can be too expensive to compute. But it can be approximated by using a subset of representatives D_c drawn randomly from $V \backslash V_c$. This idea is similar to the negative sampling approach [89], which has been successfully applied to learning the distributed representation of words in documents [90], [91]. The log-likelihood of an observed cascade then becomes

$$\mathcal{L}_c \approx \sum_{u \in V_c} \log p_{u,c}(t_u^c) + \sum_{u \in D_c} \log p_{u,c}(T) \tag{3.8}$$

where D_c is the set of negative samples chosen for every cascade by drawing random nodes uniformly from V. If we fix the size of each D_c as d, then a speedup of approximately $|V|/d$ times can be achieved.

To estimate the parameters A_u for each u and M_c for each c, we maximize the likelihood which factorizes into the product of the likelihoods of k cascades. This goal is equivalent to

maximizing the sum of the log-likelihood of K cascades

$$\{\hat{A}_u\}, \{\hat{M}_c\} = \arg\max_{\{A_u\},\{M_c\}} \sum_{c=1}^{k} \mathcal{L}_c \tag{3.9}$$

It is worth noting that the problem defined by Eq. 3.9 does not require explicit network topology to estimate A_u and M_c. Instead, the input of the model is the response times of the nodes to every cascade. This is a practical setting when the underlying network topology is incomplete or hidden during the information propagation process. In addition, the parameter space $\{A_{ui}, M_{ci} | \forall i, u, c\}$ in Eq. 3.9 does not have any restriction thanks to the adoption of the sigmoid function in Eq. 3.5.

The optimization problem in Eq. 3.9 is unfortunately not convex. Since the network size can be large, the optimization problem involves a large number of parameters, which makes stochastic updates more suitable than the batch methods. In addition, when some new cascade data comes in, the estimation algorithm should be able to incorporate the new cascades efficiently. For these reasons, we apply the Stochastic Gradient Ascent (SGA) method to estimate the parameters.

If we substitute Eq. 3.8 into Eq. 3.9, the partial derivative of the objective function F in Eq. 3.9 over a particular M_c becomes

$$\frac{\partial F}{\partial M_c} = \sum_{u \in V_c} \frac{\partial \log p_{u,c}(t_u^c)}{\partial M_c} + \sum_{u \in D_c} \frac{\partial \log p_{u,c}(T)}{\partial M_c} \tag{3.10}$$

which is a weighted sum of the terms in the form of $\frac{\partial \log p_{u,c}(t)}{\partial M_c}$. Given the value of t, $p_{u,c}(t)$ depends only on A_u and M_c. The partial derivative of $\log p_{u,c}(t)$ over M_c can be computed using A_u and M_c

$$\frac{\partial \log p_{u,c}(t)}{\partial M_c} = \big[1 - \sigma(A_u \cdot M_c) wt\big] \big[1 - \sigma(A_u \cdot M_c)\big] A_u \tag{3.11}$$

Here we need the A_u for $\forall u \in V_c \cup D_c$ to update M_c according to the gradient in Eq. 3.10. Similarly, the partial derivative of the objective function F in Eq. 3.9 over A_u can be computed using the M_c for cascades c such that $u \in V_c \cup D_c$

$$\frac{\partial F}{\partial A_u} = \sum_{c: u \in V_c} \frac{\partial \log p_{u,c}(t_u^c)}{\partial A_u} + \sum_{c: u \in D_c} \frac{\partial \log p_{u,c}(T)}{\partial A_u} \tag{3.12}$$

where the term $\frac{\partial \log p_{u,c}(t)}{\partial A_u}$ depends on A_u and M_c only

$$\frac{\partial \log p_{u,c}(t)}{\partial A_u} = \big[1 - \sigma(A_u \cdot M_c)wt\big]\big[1 - \sigma(A_u \cdot M_c)\big]M_c \tag{3.13}$$

In this way, the parameters can be updated in a pair-wise manner: we fix A_u for all u and update all the M_cs, and then fix all M_cs to update A_us in every SGA iteration. The SGA updates can operate on a bipartite graph where node u and cascade c are connected if $u \in V_c \cup D_c$ (cf. the example shown in leftmost diagram in Figure 3.5). To update A_u, each cascade c propagates the corresponding parameters M_c through the links in the bipartite graph to the node u. Then node u calculates the partial derivative $\frac{\partial \log p_{u,c}(t)}{\partial A_u}$ for every connected cascade c and updates A_u accordingly. Similarly, the nodes can propagate the parameters A_us through the links in this bipartite graph towards each relevant cascade c, and M_c can be updated using the A_u it receives.

The pseudo code is shown in Algorithm 2. The **time complexity** of each SGA iteration here is linear in the number of edges in the bipartite graph. This is because the loop in lines 8-19 iterates over all the nodes u connected with each cascade c, which corresponds to visiting every edge of the bipartite graph exactly once. Similarly, the lines 20-31 also visit every edge in the bipartite graph once. In the news dataset, the number of edges of the bipartite graph is defined by the number of news reports, because every report connects a news site to a news cascade, plus the pair of negative samples edges (u, c) for $u \in D_c$ with $|D_c| = d$ fixed as a constant. Hence, the time complexity of each SGA iteration is linear in the number of edges in the bipartite graph. The time complexity of Algorithm 2 is also determined by the number of SGA iterations. In practice, this only involves tens of iterations before the derived A_u and M_c vectors become stable. Thus, this number can be treated as a constant. Therefore, the Algorithm 2 has a linear time complexity in the number of bipartite graph edges, i.e. the number of news reports.

3.2.3 Parallelization for distributed memory machines

In practice, the input network size can be large so to speed up computation we offer parallelization of the parameter estimation for our model. Since the SGA algorithm is inherently sequential, many works including [90], [91] use the Hogwild! framework [92] attempting to parallelize the SGA algorithm on shared memory machines. The Hogwild!

Algorithm 2 SGA Algorithm using a Single Processor

α = the SGA stepsize
for each cascade c **do**
  for each node $u \in V_c$ **do**
    t_u^c = infection delay of node u in cascade c
  end for
end for
for each SGA iteration **do**
  for each cascade c **do**
    for each node $u \in V_c \cup D_c$ **do**
      if $u \in V_c$ **then**
        $t = t_u^c$
      else if $u \in D_c$ **then**
        $t = T$
      end if
      for $i = 1, 2, \ldots, m$ **do**
        A_{ui} += $\alpha \frac{\partial \log p_{u,c}(t)}{\partial A_{ui}}$
      end for
    end for
  end for
  for each node $u \in V$ **do**
    for each cascade c such that $u \in V_c \cup D_c$ **do**
      if $u \in V_c$ **then**
        $t = t_u^c$
      else if $u \in D_c$ **then**
        $t = T$
      end if
      for $i = 1, 2, \ldots, m$ **do**
        M_{ci} += $\alpha \frac{\partial \log p_{u,c}(t)}{\partial M_{ci}}$
      end for
    end for
  end for
end for

framework ignores the write-write conflicts caused by parallel updates on the same parameter as long as one unique processor can complete its writing operation. The quality of the results produced by SGA algorithm is guaranteed by the property that such conflicts are sparse enough. In the information diffusion context, however, such data sparsity is uncertain. To handle the contentions between processors, we propose a scalable parallelization scheme based on message passing paradigm [93].

The parallelization scheme is illustrated in Figure 3.5. Consider eight nodes (blue)

involved in eight cascades (yellow) in this toy example. The bipartite graph connects every pair of associated nodes and cascades in the SGA updates. These nodes and cascades are then distributed in this example to two processors. The first processor owns the upper four nodes and upper four cascades while the remaining nodes and cascades are assigned to the second processor. Each processor creates private memory space for the parameters of the nodes and cascades it owns. Much like Algorithm 2, the SGA algorithm propagates parameters back and forth between nodes and cascades, the only difference here is that the propagation between the cascades and nodes owned by different processors requires inter-core communication.

During every SGA iteration, each node u propagates its A_u to all the connected cascades in the bipartite graph. If these cascades are located in the same processor which owns node u, then this propagation operation is done locally. Otherwise, A_u is sent **asynchronously** to the processor owning node u. The pseudo code for the parallelization scheme is shown in Algorithm 3 and Algorithm 4. Without loss of generality, we consider updating M_cs using A_us, i.e. the line 14 in Algorithm 3. One SGA iteration consists of the following three phases:

- (a) Message passing: Every processor sends the A_us it owns to the target processors which require those A_us to update their corresponding M_cs, cf. the lines 1-3 in Algorithm 4.
- (b) Local updates: Every processor updates the M_c it owns using the gradients $\frac{\partial \log p_{u,c}(t)}{\partial M_c}$ if it also owns A_u, cf. the lines 4-6 in Algorithm 4.
- (c) Remote updates: After all the remote A_us sent in phase (a) have been received, each processor updates M_c using the gradients $\frac{\partial \log p_{u,c}(t)}{\partial M_c}$ whose computation requires some of the received A_us, cf. the lines 8-10 in Algorithm 4.

Note that each processor should conduct the local updates prior to the updates which require remote data from other processors, i.e. the phase (b) occurs before the phase (c). In this way, the local computation time and inter-core communication time overlap with each other, improving the parallelization efficiency.

Figure 3.5 illustrates this parallelization scheme. The three phases described above are represented by the three diagrams following the first diagram showing the initial stage in Figure 3.5. The parameters propagate back and forth between node layer and cascade layer

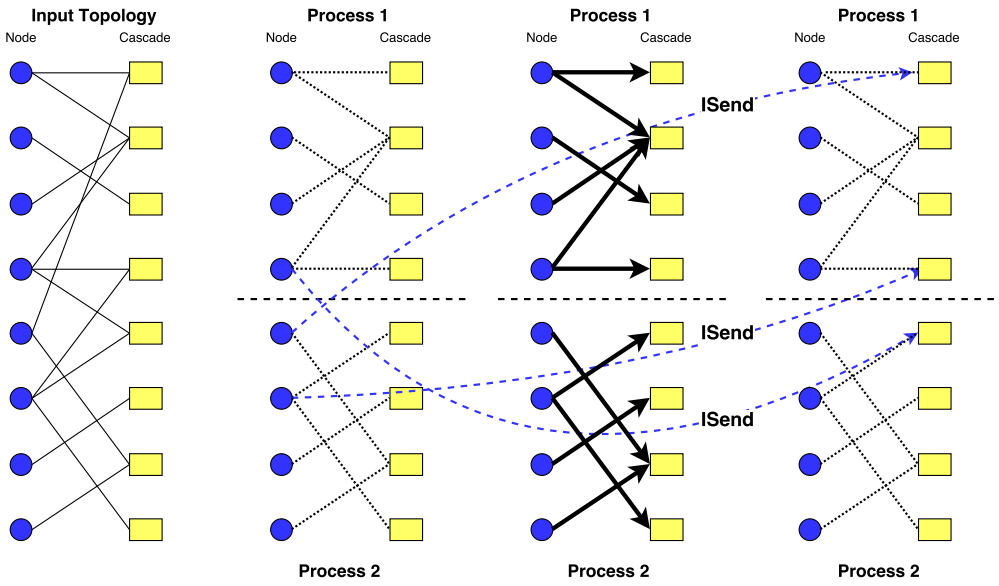


Fig. 3.5. Illustration of the parallelization scheme using two processors. The parameters propagate back and forth between nodes and cascades iteratively. If the connected node and cascade are in the different processors, the parameters are sent via asynchronous communication, i.e. the blue dashed lines marked by "ISend". At the same time, the local parameter propagation occurs within each processor, i.e. the bold black arrows in the middle. This figure illustrates the parameter propagation from node layer to cascade layer. The parameter propagation from cascade layer to node layer is conducted in a similar manner.

iteratively. If the connected node and cascade are in the different processors, the parameters are sent via asynchronous communication, i.e. the blue dashed lines labeled "ISend". At the same time, the local parameter propagation occurs within each processor, as indicated by the bold black arrows in the middle. Using the same protocol, we can update A_us using the values of M_cs.

After distributing nodes and cascades to the processors, each processor creates its private memory space for the A_us and M_cs it owns and ghost memory space for those A_us and M_cs connected to the nodes or cascades in the bipartite graph, but owned by other processors. In every SGA iteration, once the ghost memory is filled with the received data, it will not be written again. For example, a processor owns two nodes u and v, both involved in a cascade c which is owned by another remote processor. To update A_u and A_v, M_c is sent asynchronously to this local processor. But M_c will be sent only once regardless of the number of associated nodes owned by the local processor, because M_c can be shared by the updates of A_u and A_v. This optimization is similar to the combiner applied to the message

Algorithm 3 Parallelized SGA Algorithm (Distributed Memory Machines)

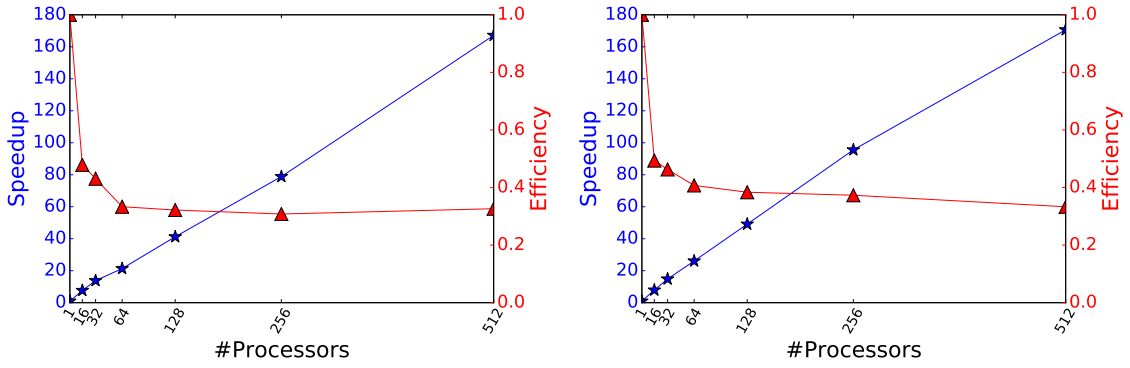
for each cascade c **do**
    $proc(c)$ = ID of the processor storing M_c
end for
for each node u **do**
    $proc(u)$ = ID of the processor storing A_u
end for
for all each processor p **in parallel do**
    U_p = IDs of the nodes owned by processor p
    C_p = IDs of the cascades owned by processor p
end for
for each SGA iteration **do**
    for all each processor p **in parallel do**
        Call Algorithm 4 to update M_cs using A_us
    end for
    for all each processor p **in parallel do**
        Call Algorithm 4 to update A_us using M_cs similarly
    end for
end for

Algorithm 4 Parallelized SGA Updates of M_cs using A_us (Distributed Memory Machines)

for each (u, c) s.t. $u \in U_p$, $c \notin C_p$ and $u \in V_c \cup D_c$ **do**
    Send A_u to $proc(c)$ asynchronously
end for
for each (u, c) s.t. $u \in U_p$, $c \in C_p$ and $u \in V_c \cup D_c$ **do**
    Call Algorithm 2 to update M_c using A_u.
end for
Wait for asynchronous receive requests until done
for each (u, c) s.t. $u \notin U_p$, $c \in C_p$ and $u \in V_c \cup D_c$ **do**
    Call Algorithm 2 to update M_c using A_u.
end for

queues in Pregal-like parallel graph processing systems [94], [95] to avoid sending duplicated messages to the same target processor.

We test our parallelization scheme on RPI Advanced Multiprocessing Optimized System (AMOS), which is a 5-rack, 5K nodes, 80K cores IBM Blue Gene/Q system [96] with additional equipment. In AMOS supercomputer, each node consists of a 16-core, 1.6 GHz A2 processor, with 16 GB of DDR3 memory. Considering the fact that the inter-core communication is generally more efficient inside the same node than across different nodes, we use all the 16 cores of a node so that the communication between cores can be more efficient.



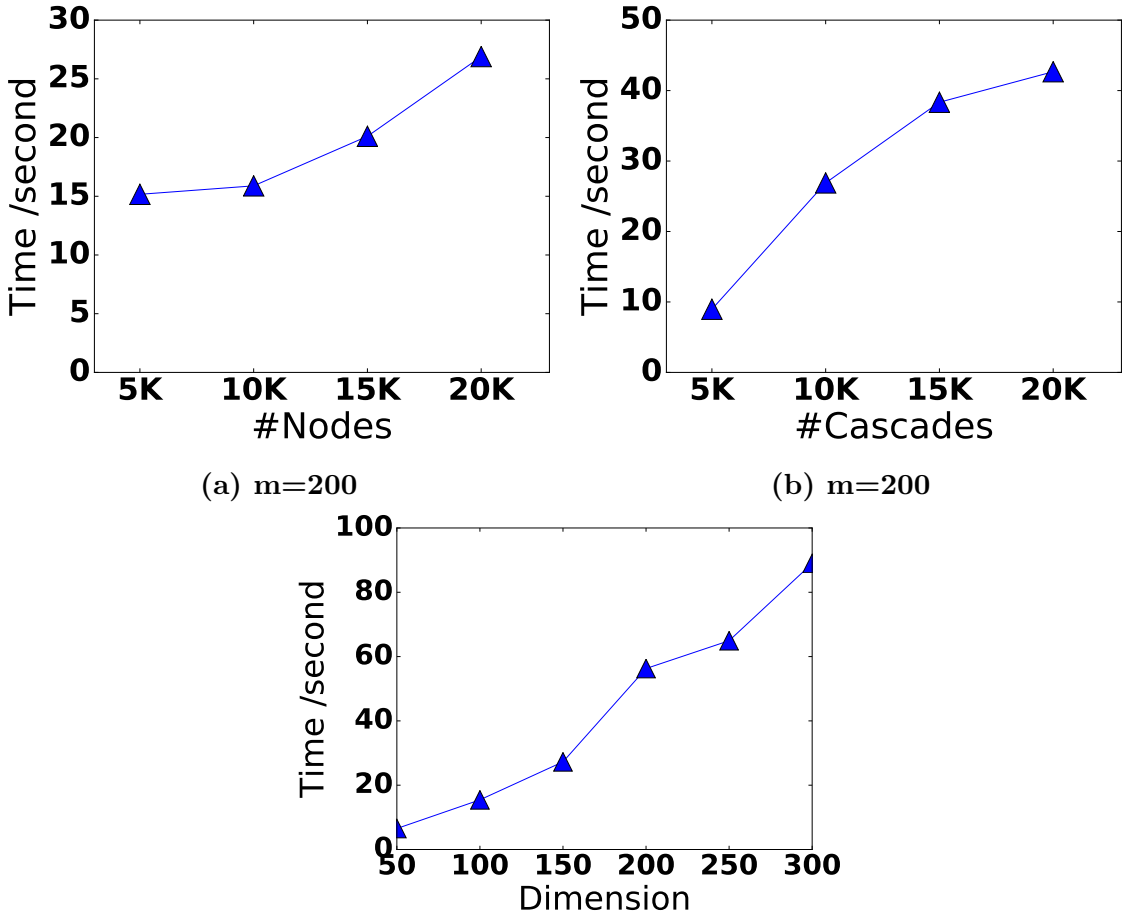
(a) m=100 **(b) m=200**

Fig. 3.6. Speedup and efficiency of our parallelization scheme on Advanced Multiprocessing Optimized System (AMOS). m denotes the dimension of A_u and M_c.

In general, the maximum number of cores per node here is not constrained by the limit of 16 GB DDR3 memory space. This is one benefit of our memory management paradigm because every processor stores only one copy of the parameters of nodes and cascades associated with it.

The speedup and efficiency of our parallelization scheme are shown in Figure 3.6. A total of 10K cascades are simulated on the SBM network with 20K nodes. Each cascade infects a total of 247 nodes on average. The Figure 3.6 shows that the parallelization scheme achieves an approximately linear speedup using several hundreds of processors. And the efficiency of the parallelization scheme is above 25% in all cases. The comparison between Figure 3.6a and Figure 3.6b shows that the dimension m does not change the speedup or efficiency. In our sampled GDELT dataset, the parallelized algorithm achieves the similar speedup and efficiency using 64 processors, but adding extra processors does not significantly increase the speedup due to the limited size of the dataset.

To test the scalability of the parallelization scheme, we evaluate the execution time of one single SGA iteration of Algorithm 3 using 512 processors. Given a network of 10K nodes and the different numbers of cascades, Figure 3.7a shows that the execution time is approximately proportional to the number of cascades. A similar pattern is also observed as the dimension m grows. In contrast, when the number of cascades is fixed and the number of network nodes increases, the execution time grows slowly - the execution time only doubles when the number of network nodes grows from 5K to 20K. The reason is that the number of infections per cascade is relatively stable in the synthetic data so that the increase of time



(a) m=200 (b) m=200

(c) m=200

Fig. 3.7. The execution time of one SGA iteration in relation to the dimension m, the number of network nodes and the number of cascades.

for parameter propagation is actually smaller than the increase in the number of network nodes. In general, the parallelization scheme scales well with the number of network nodes, the number of cascades and the dimension m.

3.2.4 Virality prediction via early adopters

Our aim is to forecast the viral information cascade. From the historical cascades, the proposed model estimates the A_u vector for each u according to the observed response times. Using these A_u vectors of the initially infected nodes, we seek to predict the behavior of future cascades.

Suppose a set of so-called early adopters have been infected within a limited time period. One basic observation is that, once the contagion reaches a community member, the

Table 3.1. Accuracy of the news virality prediction measured by F1 scores. The predictions of our proposed model (ML) are consistently better than the the baseline model's (BL) using the same set of early adopters in the first 30 to 60 minutes.

Threshold θ	τ =0.5 Hour		τ =0.6 Hour		τ =0.7 Hour		τ =0.8 Hour		τ =0.9 Hour	
	BL	ML	BL	ML	BL	ML	BL	ML	BL	ML
90%	0.410	0.500	0.410	0.497	0.410	0.499	0.499	0.549	0.499	0.548
91%	0.412	0.484	0.405	0.486	0.405	0.480	0.484	0.534	0.490	0.536
92%	0.389	0.473	0.394	0.472	0.392	0.471	0.463	0.530	0.463	0.531
93%	0.294	0.455	0.296	0.453	0.294	0.453	0.434	0.516	0.436	0.511
94%	0.207	0.433	0.209	0.433	0.213	0.436	0.393	0.489	0.393	0.491
95%	0.191	0.404	0.191	0.409	0.195	0.408	0.352	0.465	0.347	0.462
96%	0.147	0.393	0.141	0.389	0.147	0.389	0.287	0.438	0.285	0.442
97%	0.114	0.360	0.116	0.366	0.113	0.361	0.233	0.402	0.232	0.407
98%	0.097	0.311	0.091	0.316	0.097	0.316	0.160	0.365	0.166	0.357
99%	0.119	0.288	0.107	0.292	0.104	0.279	0.148	0.326	0.148	0.324

probability that other members get infected increases. Therefore, we can make use of the infected node's local neighborhood to predict the infection future. Since our model presents every node u by a vector A_u in the latent space, it is easy to find the neighbors which are close to node u by measuring the Euclidean distances between them. As the contagion is likely to spread fast in the dense areas, the number of neighbors within a certain range can be used as an indicator of future infections. Hence, we count the number of neighbors that are located within a certain range from the infected node v, and arrange these values in a vector K whose i-th component is defined as

$$K_i = |\{u | \|A_u - A_v\|_2 < r_i\}| \tag{3.14}$$

where r_i is the radius of the i-th neighborhood of node v and $\|\cdot\|_2$ denotes the Euclidean norm.

Figure 3.8 demonstrates how to compute the K vector of an infected node. Suppose the infected nodes are located at the centers of the different circles. Their neighborhoods are marked by the circles which have the radii r_1, r_2 and r_3 respectively. In Figure 3.8, the node at the right center has 3 neighbors inside the small circle, 7 neighbors inside the medium circle and 12 neighbors inside large circle, resulting in the vector $K = [3, 7, 12]$. The leftmost node has only 1 neighbor, i.e. itself, inside both the small and medium circle and

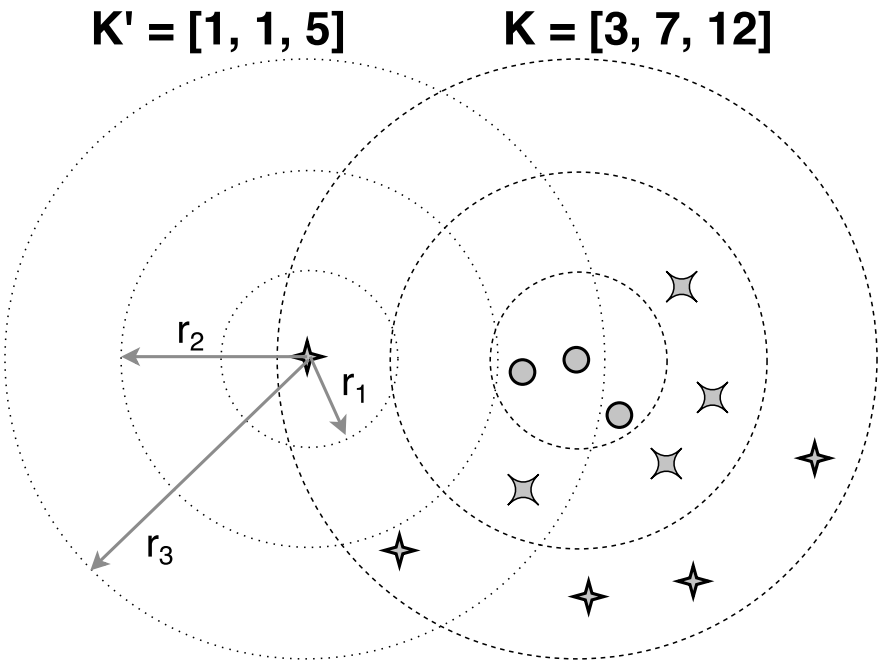


Fig. 3.8. Illustration of the local neighborhoods in the vector space. The concentric circles mark the neighborhoods with different radii from the center. The proposed method counts the number of nodes located in each circle and arrange these values in a vector. Vector K shows the numbers of nodes for r_1, r_2 and r_3 radii drawn from the figure center while K' is shown for the circles centered at the asterisk on the left.

5 neighbors inside the large circle, resulting in the vector $K' = [1, 1, 5]$. Intuitively, we can tell from K and K' that the right three neighborhoods allow faster growth of infections than the left ones.

Given a set of early adopters which get infected within a limited time period, we can count the total number of **unique** neighbors in their local neighborhoods within different radii similarly. Note that if a node is in the i-th neighborhood of two early adopters, this node is only counted once in the i-th component of K. Finally, The numbers of neighbors, presented as the components of a multiple-dimensional vector K, are fed to a machine learning model to predict the final size of a cascade.

Our model produces the A_u vector for each node u in the network. If these $\{A_u\}$ vectors preserve the community structures of the news media network well, their clustering should match the community structure embedded in the explicit network topology, because the members of a community have similar A_us.

Given a synthetic SBM network, we simulate the cascades as described in Section 3.2.1.

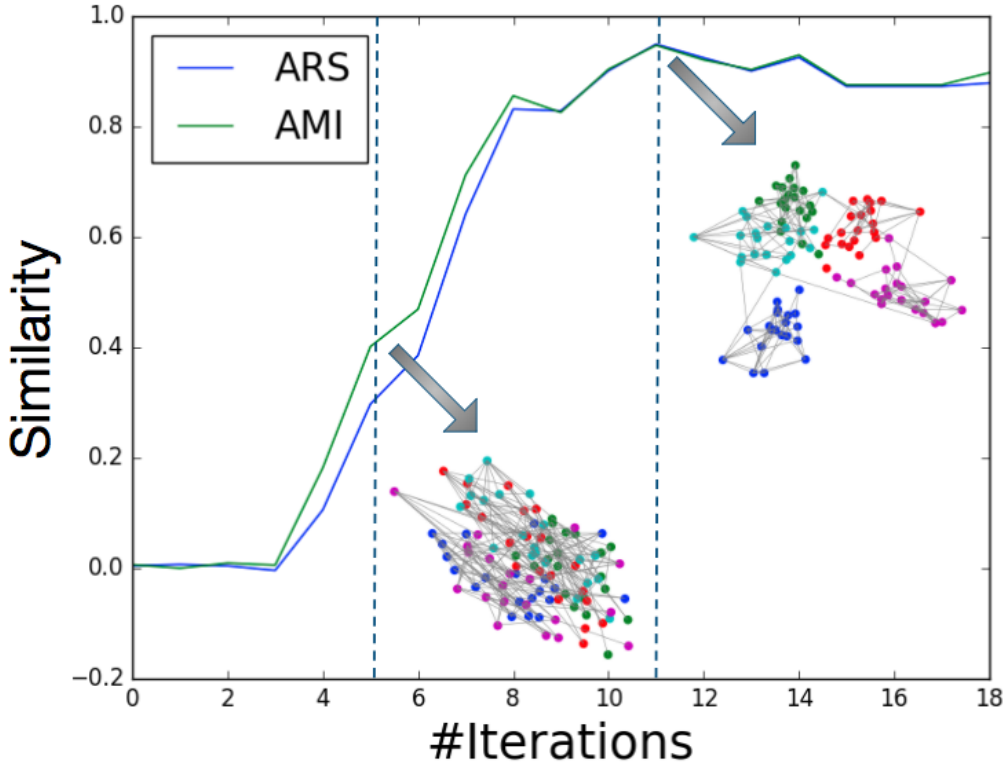


Fig. 3.9. The similarity between the node clustering of the $\{A_u\}$ vectors at each SGA iteration of Algorithm 2 and the ground truth partition of the SBM network.

Our model then infers the $\{A_u\}$ vectors by these cascades. For clarity, the set of nodes detected in a network is called a community and the set of nodes clustered by their vector representations is called a cluster here. We compare the node clustering of these $\{A_u\}$ vectors with the ground truth partition of the SBM network and the community structures discovered by traditional community detection algorithms from the explicit topology. The alignment between them indicates the $\{A_u\}$ vectors produced by our model preserve the community structures.

More specifically, the K-means clustering algorithm [97] is executed on the so-inferred $\{A_u\}$ vectors to derive the node clustering. The similarity between the node clustering and the contrastive partitioning of the network are measured by the Adjusted Mutual Information (AMI) and Adjusted Rand Score (ARS) which are widely used to evaluate community detection performance. Section 2.2.3 presents the mathematical definitions of both metrics.

Figure 3.9 shows the growth of the similarity between the node clustering of the $\{A_u\}$ vectors at each SGA iteration and the ground truth partition of the SBM network. The

Table 3.2. The pairwise similarities between the communities detected by the state-of-the-art community detection algorithms, the node clustering of the $\{A_u\}$ vectors produced by our model and the ground truth partition of the SBM network. The entries below and above the main diagonal represent the Adjusted Rand Score (ARS) and Adjusted Mutual Information (AMI) respectively. FG: Fast Greedy algorithm [98], LE: leading eigenvector method [56], LP: label propagation algorithm [64], ML: multilevel algorithm [51]. Our model produces node embeddings whose clustering aligns well with the ground truth communities, even outperforming some community detection baseline methods.

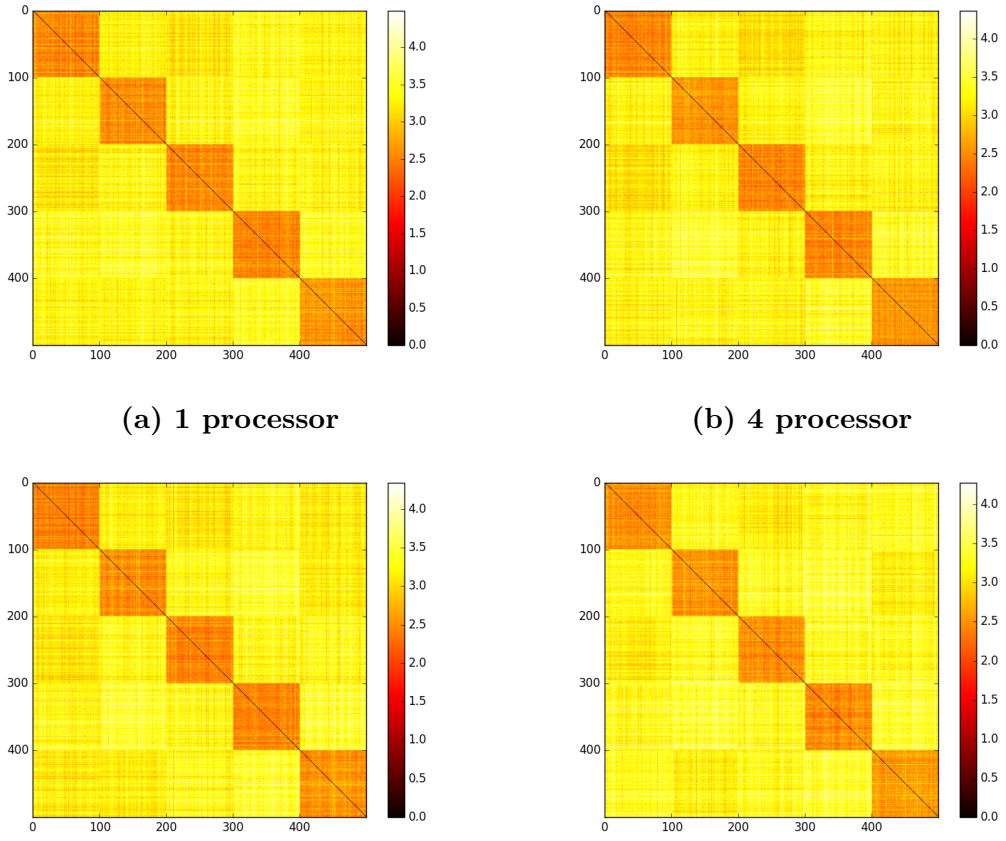
ARS / AMI	FG	LE	LP	ML	Our Model	Ground Truth
FG		0.858	0.833	0.943	0.881	0.933
LE	0.873		0.807	0.867	0.795	0.837
LP	0.864	0.829		0.835	0.773	0.804
ML	0.929	0.881	0.868		0.922	0.949
Our Model	0.869	0.820	0.817	0.936		0.930
Ground Truth	0.939	0.865	0.852	0.963	0.925	

SBM network has 100 nodes and 5 communities of size 20, 190 edges connects nodes in the same community and 12 edges are across different communities. A total of 100 cascades are simulated, each involves 12.5 infections on average. The dimension of A_u is $m = 10$. After each SGA iteration of Algorithm 2, we compute a 100×100 distance matrix with the (u, v) entries being the Euclidean distances between the latest updated vectors A_u and A_v. The plots for two networks in Figure 3.9 are made by the MDS algorithm [99] which places each node in two-dimensional space such that the derived between-node distances are preserved as well as possible. In other words, the MDS coordinates preserve the nodes' pairwise distances in the high-dimensional space of $\{A_u\}$. At the 5th iteration, the ARS and AMI scores are around 0.4, the nodes' MDS coordinates do not reflect their ground truth communities represented by the color. When the 11th SGA iteration is done, the ARS and AMI scores become greater than 0.9, and the nodes' MDS coordinates match the community structures very well. Figure 3.9 shows that, as the inference algorithm proceeds, the $\{A_u\}$ vectors start to preserve the community structures, even though our model takes only the infection delays in the cascades as input, but not the the explicit network topology.

In addition, we compare the node clustering of $\{A_u\}$ vectors at the 15th iteration with the ground truth partition of the SBM network and community structures detected by the state-of-the-art algorithms such as Fast Greedy algorithm [98], leading eigenvector

Table 3.3. Quality metric of the detected communities on synthetic SBM networks with 10K nodes. ARS: Adjusted Rand Score; AMI: Adjusted Mutual Information.

#Processors	**1**	**4**	**16**	**64**
ARS	0.9588	0.9480	0.9444	0.9704
AMI	0.9858	0.9814	0.9808	0.9888



(a) 1 processor **(b) 4 processor**

(c) 16 processor **(d) 64 processor**

Fig. 3.10. Distance matrix of the first 500 nodes of synthetic SBM networks based on the node embeddings produced by different number of processors. The color in the distance matrix indicates the distance between a pair of nodes, brighter the color longer the distance.

method [56], label propagation algorithm [64] and multilevel algorithm [51]. Table 3.2 shows that the alignments between them are good, and that the node clustering of $\{A_u\}$ vectors are more similar to the ground truth partition than the community structures detected by some state-of-the-art community detection algorithms. In Table 3.2, each entry indicates the similarity of the communities produced by a particular pair of methods. All the entries

below the main diagonal correspond to the ARS scores and entries above correspond to the AMI scores. As the ARS and AMI scores indicate, our model produces meaningful node embeddings because the clustering of these node vectors aligns well with the ground truth communities. The community structure obtained by clustering node vectors is even closer to the ground truth than are the community structures detected by the baseline methods that include leading eigenvector method (LE) and label propagation algorithm (LP) are. Our model produces node embeddings whose clustering aligns well with the ground truth communities, outperforming some community detection baselines. In addition, our model does not use the topology of the SBM network like the baseline algorithms do, instead it only accesses the cascades data, which explains why Fast Greedy algorithm (FG) and multilevel algorithm (ML) performs better than our model in terms of community detection. Finally, it should be noted that we choose the number of clusters as 5 for the K-means algorithm here. However this number should be actually systematically selected. We leave the selection of the proper number of clusters for future work.

We also test our algorithm for large SBM networks with the dimension of resulting A_u being 200. In these experiments, there are 100 predefined communities in the SBM network, each containing 100 nodes. Every node is connected to 8.8 nodes in the same community and 1.2 nodes in the other communities on average. And we simulate 10K cascades using the continuous time IC model. As shown in Table 3.3, as the number of processors increases, the AMI and ARS metrics are consistently above 0.98 and 0.94 respectively, which indicates the resulting $\{A_u\}$ preserves the community structure in the network. The distance matrices of the first 500 nodes are also shown in Fig. 3.10. In the distance matrix, the distance between two nodes u and v is defined as the Euclidean distance between vectors A_u and A_v and this value is visualized by the color brightness in the heatmap, brighter the color longer the distance. Each dense module in this matrix comprises 100 nodes and matches the predefined SBM community very well. As illustrated by the visualized pair-wise nodes distance matrices, the number of processors does not change the high quality of $\{A_u\}$ as the resulting vectors preserve the community structure of SBM networks in all cases. Our model does not use the topology of the SBM network, instead it only accesses the response times of the nodes to different cascades, yet the community structure can still be accurately recovered from these response times.

Our aim is to predict the viral news cascades at their early stage. Specifically, with

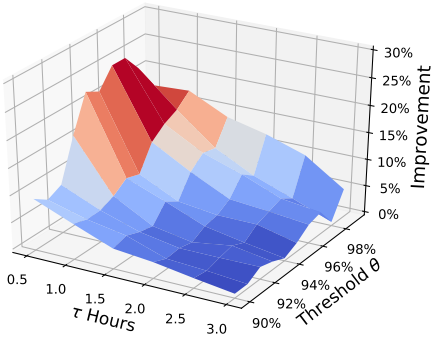
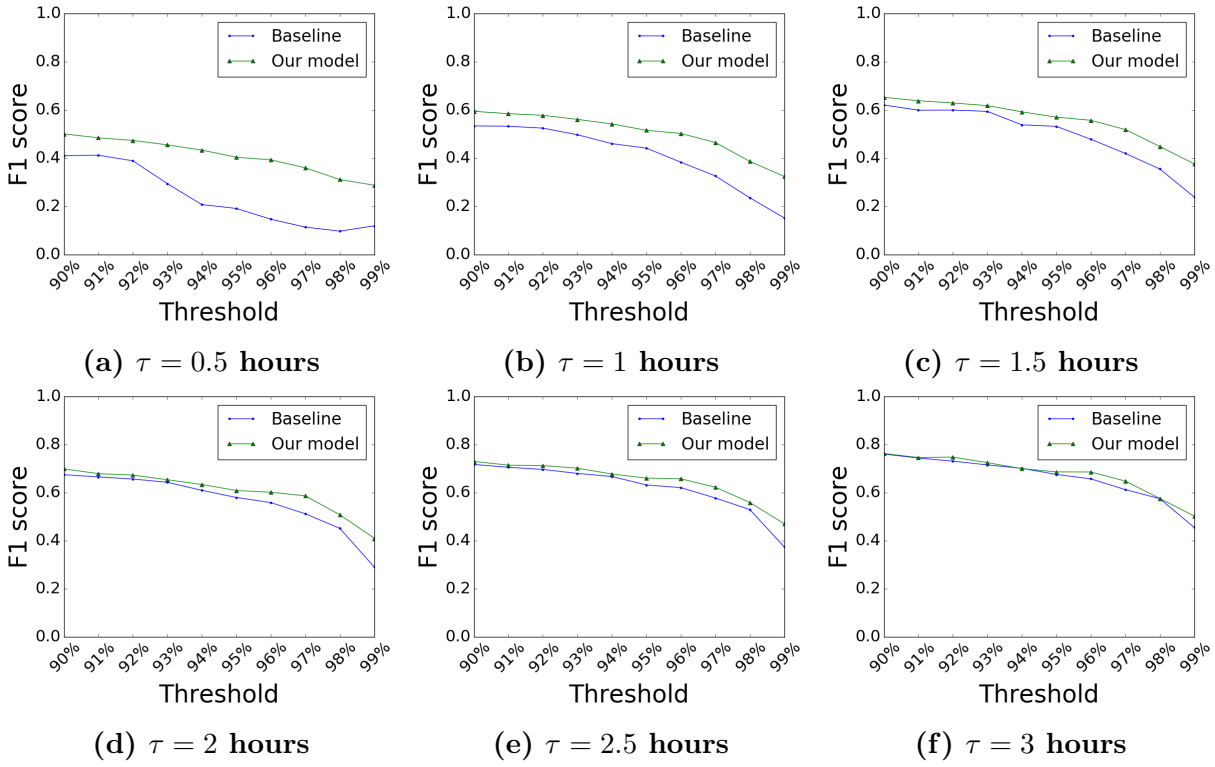


Fig. 3.11. The improvement in virality prediction accuracy produced by our model in GDELT dataset in relation to the classification threshold θ and the initial observation period of τ hours.

the global news dataset of events, the task is to predict the most reported events within a limited time period. Therefore, we rank the events reported in the news by the number of their reports and divide them into two classes: those who are among the top $(1-\theta)$ percent of this ranking and the remaining events. In this way, we can treat the virality prediction as a binary classification problem - given the early reports within a limited time period, can we classify the events reported in the news into these two categories? Since we are only interested in predicting the most viral events reported in the news, the threshold θ ranges from 90% to 99% in our experiments. Notice that a high threshold θ would result in two very imbalanced sets of samples, which would make the prediction challenging.

Baseline We build a baseline algorithm which uses multiple features extracted from cascade early progress and the Random Forest model [100] for cascade classification. We choose Random Forest for comparison because, as an ensemble learning method, it is known to work well with the non-linear growth of modeled phenomena such as the viral spread of news reports. The extracted features include the number of unique early adopters, the frequency of the infections at the early stage, the maximum interval between two continuous infections, and the minimum interval between two continuous infections. In contrast, our proposed model uses the numbers of neighbors in different ranges from the infected nodes, i.e. vector K presented in Eq. 3.14, as the input of the Random Forest model. For a fair comparison, both the baseline and our model use the information about the early adopters in the first τ hours, where τ ranges from 0.5 to 3.



(a) $\tau = 0.5$ **hours** **(b)** $\tau = 1$ **hours** **(c)** $\tau = 1.5$ **hours**

(d) $\tau = 2$ **hours** **(e)** $\tau = 2.5$ **hours** **(f)** $\tau = 3$ **hours**

Fig. 3.12. The accuracy of virality prediction in global news dataset of events (i.e. GDELT) measured by the F1 score. The prediction models take the news sites reporting an event in the first τ hours, i.e. the early adopters, as input.

The accuracy of the prediction is evaluated by the F1 score which is commonly used in the classification problems

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3.15}$$

F1 score considers both the precision and recall of virality prediction. A decent F1 score prevents the system from either predicting too many viral news with a high false positive rate, or from being too conservative making insufficient predictions.

Figure 3.12 shows the F1 scores of the 6-fold cross validation tests using a variety of τ values. As the threshold θ increases, the F1 scores of both the baseline and our model decrease due to the imbalanced sets of samples. The F1 scores of the prediction made by our model are consistently better than the baseline's. Specifically, our model outperforms the baseline by 10% in most cases. As shown in Figure 3.11, the improvement produced by our model is very obvious when the value of τ becomes small. It is because our model uses the community structure of the propagation network which is not included in the feature-

based baseline model. As the value of τ increases, both the baseline model and our proposed model gain better performance, yet the performance gap between them decreases at the same time. One potential reason is that the information cascades start to slow down inside each communities at this stage with $\tau > 2$. Thus, the community structure does not provide extra signal for the prediction as it does at the early stage with $\tau < 1.5$. In general, in terms of the prediction accuracy, the influence of τ value is more significant in the baseline model than in our model, which indicates that community structures can provide the critical signals to forecast the viral information cascades at the early stage.

The relationship between the improvement on prediction accuracy and the classification threshold θ is shown in Figure 3.11. Our proposed model performs much better than the baseline model when the threshold is high, and it achieves an almost 25% improvement with the threshold $\theta = 98\%$. As discussed in Section 3.2.4, our proposed method calculates the number of neighbors in the early adopters' the local neighborhood, i.e. the number of nodes whose A_u vectors are close to the early adopters' in the latent space. Here, the most plausible explanation is that the most viral cascades have the early adopters in multiple dense areas so that they have advantages in disseminating the contagion to their neighbors in these regions in parallel, resulting in the viral infections within a limited time period. This explanation matches our observation about the viral news in online media - most news about events rarely cross the geographical and cultural boundaries, but once they do, the breaking news draw attention from the news media sites in different regions and hit the headlines very quickly.

3.3 Scale-free peer-to-peer network construction

The scale-free property is shown to exist in many natural or artificial complex systems, such as protein-protein interaction networks [101], the Internet [102], the World Wide Web [103], and scientific collaboration networks [104]. The degree distribution in these networks follows the power-law: $P(i) \sim i^{-\gamma}$, where $P(i)$ is the fraction of nodes with degree i and γ is the scaling parameter which varies between different types of networks ($2 \leq \gamma \leq 3$ in most cases). In a limited scale-free topology, only nodes with degrees smaller than the hard cut-off (i.e. the maximum) degree have degree distribution that follows the power-law.

The scale-free topology has some good properties, including high tolerance to random attacks [76], high synchronizability [77] and resistance to congestion [78]. For this reason,

several growth models are proposed to construct the scale-free overlay topology. The BA model [79] manages to explain the evolution of scale-free topologies by a core principle named "Preferential Attachment". But it is not practical in real distributed applications because the global information is required to maintain it. To address this issue, HAPA [80], Gaian [81] and SRA [82] algorithms were introduced to construct the scale-free overlay topology with partial or no global information.

"Preferential Attachment" [79] means a new node is more likely to connect to heavily linked nodes when it joins the network. The BA model has some disadvantages as the growth model for the overlay topology. Firstly, it does not provide hard cut-offs. Since a heavily linked node uses a lot of bandwidth, nodes usually are not willing to maintain high degrees. For this reason, a user-defined hard cut-off (i.e. the maximum) degree is imposed lower than the natural cut-off arising in the BA model. The imposed hard cut-offs restrict the feasible overlays to the limited scale-free topologies, which are more practical. Moreover, the BA model also requires the global information about the topology to add connections when a new node joins. In real-world applications, however, the communication cost of obtaining the global information is prohibitive. Therefore, a distributed approach that constructs the topology without global information is desired.

More formally, the degree of a node is defined as the number of connections it has in an overlay topology. Using the same notation as in [82], the fraction of nodes with degree i is denoted as P_i,

$$P_i = \frac{N_i}{N} \tag{3.16}$$

where N_i is the number of nodes with degree i and N is the total number of nodes.

In a scale-free topology, degree distribution follows the power-law: $P_i \sim i^{-\gamma}$ where γ is a constant. In a limited scale-free topology, P_i follows the power-law for $i < m$, where m is the hard cut-off (i.e. the maximum) degree.

The value of P_i in a limited scale-free topology is given by Eq. 7 in [82] as f_i as a function of the maximum degree m, the minimum degree k and the scaling parameter γ,

$$f_i = \frac{m - 2k}{i^{\gamma} \sum_{j=k}^{m-1} \frac{m-j}{j^{\gamma}}} \quad \text{for} \quad i < m \tag{3.17}$$

and f_m, that does not need to follow the power-law distribution, is given as,

$$f_m = 1 - \sum_{i=k}^{m-1} f_i \tag{3.18}$$

The goal is to maintain the degree distribution as f_i for $i = k, \ldots, m$ while nodes with arbitrary degrees are added or removed. For the sake of simplicity, we assume one node is removed at a time and the node to be removed is denoted as node R and the number of its neighbors is denoted by b.

3.3.1 Protocol design for dynamic topology construction

In [80], authors propose Hop-and-Attempt Preferential Attachment (HAPA) algorithm [80] where a new node joining the network connects to k (i.e. the minimum degree) nodes in a random route. This scheme works because high degree nodes are more likely to occur in a random route than nodes with low degree. In Gaian algorithm [81], a new node broadcasts a message when it joins the network using computing with time principle [105]. Each receiver computes the maximum time of delay, t_v, which is proportional to the inverse of its degree, and chooses the time of delay t_d in the interval $[0, t_v]$ uniformly randomly. Instead of replying to the sender instantly, the receiver waits the time of delay t_d and then replies. In this way, nodes with higher degrees are likely to wait shorter period. The new node connects to the first k responders. It gives a better chance to the new node to connect to nodes with high degrees. This mechanism, which reduces the communication overhead by allowing nodes to self-select themselves according to their fitness to the desired property, is also known as computing with time [105]. The communication cost for selection is constant in the number of candidates. Similar to HAPA algorithm, Gaian algorithm cannot produce an overlay topology with the user-defined scaling parameter. In addition, post-construction parameters of network structures (i.e. the scaling parameter related to the search performance) cannot be adjusted in these approaches. In [82], the authors propose a flexible growth model that can produce a limited scale-free topology with user-defined parameters. The **S**emi-**R**andomized Growth **A**lgorithm (SRA) [82] requires no global information and imposes a hard cut-off on degrees. In SRA, when a node joins the network, it broadcasts a message containing the desired degrees of the k new neighbors. These degrees are computed according to the given network parameters. The receivers with the desired degrees reply to the new node

using computing with time rule [105]. The new node connects to the first k responders. The scaling parameter, as well as the hard cut-off, can be defined by users in advance. So, the SRA model is able to produce overlay topologies, over which the efficiency of applications such as search algorithms is maximized. The constraints of feasible values of these network parameters are presented in [106]. SRA outperforms other growth models by producing overlay topologies with perfect matching to the arbitrary power-law degree distribution.

To the best of our knowledge, however, allowing nodes to leave network during overlay maintenance has not been studied. Yet, in peer-to-peer (P2P) networks, nodes are likely to join and leave the network frequently. With such dynamic peer participation, the scale-free topology produced by previous growth models are affected by node removal, especially when hubs are removed. This effect can accumulate, negatively impacting the performance of the P2P network, which are built on top of the overlays. We propose a robust model, in which nodes are allowed to leave the network in an arbitrary pattern. Combined with the growth model proposed in our previous work [82], the scale-free topology is maintained while nodes are allowed to freely join and leave the network. We present our approach in detail below.

We propose the following two types of atomic operations that can be conducted by each neighbor of R, which is the single node removed from the network:

- PUSH: The neighbor of R connects to a new node A.
- SHUFFLE: Besides connecting to a new node, the neighbor of R also asks the new node A to terminate one connection to some node B.

We are interested in the degree of every PUSH and SHUFFLE when a single node is removed. Let D_i denote the number of SHUFFLEs on degree i and I_i denote the number of PUSHes on degree i. Since $(b-k)$ PUSHes and k SHUFFLEs are needed, we have,

$$\begin{aligned} \textstyle\sum_{i=k}^{m-1} I_i &= b-k \\ \textstyle\sum_{i=k+1}^{m} D_i &= k \end{aligned} \tag{3.19}$$

These SHUFFLEs and PUSHs make D_i nodes *decreasing* their degree from i to $(i-1)$ and I_i nodes *increasing* their degree from i to $(i+1)$. Also, $I_m = 0$, $D_k = 0$ because the degrees of all nodes are kept in range $[k, m]$. I_i, D_i are non-negative for $i \in [k, m]$.

Consider the total number of nodes with degree k after node R is removed from a network of size n. Before removal, there were $(f_k \cdot n)$ nodes originally of degree k. Additional

D_{k+1} nodes originally with degree $(k+1)$ are added and I_k nodes are moved from this count by SHUFFLEs and PUSHes. If the fraction of nodes with degree k is still f_k, we have,

$$f_k(n-1) = f_k n - I_k + D_{k+1} \tag{3.20}$$

where n is the total number of nodes before R quits. Similarly, the degrees of I_d nodes increase from d to $(d+1)$. The degrees of I_{d-1} nodes increase from $(d-1)$ to d. The degrees of D_{d+1} nodes decrease from $(d+1)$ to d. And the degrees of D_d nodes decrease from d to $(d-1)$. If the fraction of nodes with degree d remains f_d, then

$$f_d(n-1) = f_d n - I_d + I_{d-1} + D_{d+1} - D_d \tag{3.21}$$

for $k < d < m$ and $d \neq b$. Since node R itself is removed,

$$f_b(n-1) = f_b n - I_d + I_{d-1} + D_{d+1} - D_d - 1 \tag{3.22}$$

Due to the hard cut-off, nodes with degree m should not accept any new connections,

$$f_m(n-1) = f_m n + I_{m-1} - D_m \tag{3.23}$$

Simplifying Eqs 3.20, 3.21, 3.22, 3.23, we obtain that for $k \leq i \leq b-1$,

$$I_i - D_{i+1} = \sum_{j=k}^{i} f_j \tag{3.24}$$

and for $b \leq i < m$,

$$I_i - D_{i+1} = \sum_{j=k}^{i} f_j - 1 \tag{3.25}$$

Let non-negative vectors $\vec{D} = (D_{k+1}, D_{k+2}, \ldots, D_m)^T$, and $\vec{I} = (I_k, I_k, \ldots, I_{m-1})^T$ be such

that,

$$\vec{I} - \vec{D} = \begin{bmatrix} f_k \\ f_k + f_{k+1} \\ \vdots \\ \sum_{i=k}^{b-1} f_i \\ \sum_{i=k}^{b} f_i - 1 \\ \vdots \\ \sum_{i=k}^{m-1} f_i - 1 \end{bmatrix} \tag{3.26}$$

with the L^1 norm $\|\vec{I}\|_1 = b - k$, $\|\vec{D}\|_1 = k$. The solution to Eq. 3.26 depends on the degree distribution f_i, the degree of removed node b and the hard cut-off m, but is independent from the current network size n. It allows us to design an algorithm that does not need any global information.

One simple solution to Eq. 3.26 is,

$$I_i^* = \begin{cases} 1 & k \leq i < b \\ 0 & \text{otherwise} \end{cases} \tag{3.27}$$

and for $i \in [k+1, m]$,

$$D_i^* = 1 - \sum_{j=k}^{i-1} f_j \tag{3.28}$$

It could be observed that $D_{i+1}^* = a(i)$ which is the average number of nodes increasing degree from i to $(i+1)$ when a node joins the network in the growth model [82]. This is because, intuitively, the decreasing degree is exactly the opposite to a new node's connecting to k neighbors.

According to the analysis above, there should be I_i neighbors of R that PUSH on degree i and D_i neighbors of R that SHUFFLE on degree i, for $i = k, \ldots, m$. For the specific protocol design, we use the solution I_i^* and D_i^* presented in Eqs 3.27 and 3.28.

Before node R voluntarily quits, it can compute I_i^* and D_i^* locally and assign either the PUSH operation or the SHUFFLE operation to each of its neighbors. The neighbors will PUSH or SHUFFLE as assigned by R before it quits.

If node R crashes due to errors or attacks, it can assign its neighbors the associated

operations before the failure. Since the value of I_i^* depends on the degree of R, R should re-compute I_i^* and assign the new PUSH/SHUFFLE operations to the neighbors if its degree changes. So, every node whose degree changed sends *updates messages* with its new degree and PUSH and SHUFFLE operations to its neighbors.

If the degree of a node R changes to be b', R computes two non-decreasing sequences,

$$v_i = \sum_{j=k+1}^{i} \frac{D_j^*}{k} \quad \text{and} \quad u_i = \sum_{j=k}^{i} \frac{I_j^*}{b'-k} \tag{3.29}$$

and generates two random values r_1, r_2 distributed uniformly over the range $[0, 1)$ for each of the b' neighbors. If $r_1 < k/b'$ and r_2 is in the interval $[v_i, v_{i+1})$, then the neighbor SHUFFLEs on degree i; if $r_1 \geq k/b'$ and r_2 is in the interval $[u_i, u_{i+1})$, then the neighbor PUSHes on degree i. In this way, the probability of PUSH on degree i is $(1 - \frac{k}{b'})\frac{I_i^*}{b'-k} = \frac{I_i^*}{b'}$ and the probability of SHUFFLE on degree i is $\frac{k}{b'}\frac{D_i^*}{k} = \frac{D_i^*}{b'}$.

Furthermore, node R sends the list of its neighbors' IPs in the *update message*. When R is removed from the network, all neighbors of R broadcast the last *update message* sent by R. Any receiver with one of the desired degrees in the *update message* connects to the corresponding neighbor of R. And each neighbor of R connects only to the first responder. It is worth noting that only one broadcast is needed because all neighbors of R broadcast an identical message and nodes in the network forward the first message they receive.

Applying the solution from Eqs. 3.27 and 3.28 to our protocol, we obtain a new Enhanced Semi-Randomized Growth Algorithm (E-SRA):

Approach: E-SRA (Assuming one node is removed at a time) To remove a node R with degree b, each neighbor of R either SHUFFLEs on degree i by the probability $\frac{1-\sum_{j=k}^{i-1} f_j}{b}$ for $i = k+1, \ldots, m$ or PUSHes on degree j with probability $\frac{1}{b}$ for $j = k, \ldots, b-1$.

The pseudo-code is listed as Algorithm 5. The message complexity of the algorithm is one broadcast per removal.

In Eq. 3.28, the greater i is, the smaller D_i^* is. This fact makes the protocol more practical because there are a large number of nodes with low degrees and only a small fraction of nodes with high degrees in a scale-free network. Nodes with low degrees are easily reached by broadcasts in a few hops; Eq. 3.27 implies that the number of PUSH operations increases

Algorithm 5 Enhanced Semi-Randomized Growth Algorithm (E-SRA)

```
1: procedure OnQuit
2:     Quit
3: end procedure
4: procedure OnNeighborQuit
5:     Broadcast the latest update message received from the quitting neighbor
6:     Connect to the first responder
7: end procedure
8: procedure OnReceiveUpdateMessage(updateMsg)
9:     b ← the degree of itself
10:    for each op_i in updateMsg do
11:        if op_i = ( IP_i , PUSH on degree b) then
12:            Reply to the node with IP_i
13:            Return
14:        else if op_i = ( IP_i , SHUFFLE on degree b) then
15:            Terminate its existing connection to a random neighbor RN
16:            Ask RN to reply to the node with IP_i
17:            Return
18:        end if
19:    end for
20: end procedure
21: procedure OnDegreeChange(newDegree)
22:    b ← newDegree
23:    update message ← {}
24:    for i = 1 to b do
25:        r_1 ← random number in [0, 1)
26:        r_2 ← random number in [0, 1)
27:        if r_1 < k/b then
28:            for j = k + 1 to m do
29:                if r_2 ∈ [v_j, v_{j+1}) then
30:                    op_i ← ( IP_i , SHUFFLE on degree j)
31:                end if
32:            end for
33:        else
34:            for j = k to m − 1 do
35:                if r_2 ∈ [u_j, u_{j+1}) then
36:                    op_i ← ( IP_i , PUSH on degree j)
37:                end if
38:            end for
39:        end if
40:    end for
41:    update message ← {op_1, op_2 . . . , op_b, b}
42:    Send the update message to every neighbor.
43: end procedure
```

Table 3.4. Comparison of growth models.

Algorithm	Global knowledge used	Flexible γ	Tolerance to removal
BA	Complete	No	No
HAPA	Partial	No	No
Gaian	None	No	No
SRA	None	Yes	No
E-SRA	None	Yes	Yes

linearly with the degree of node R.

Note that we assume the neighbors of node R are able to PUSH or SHUFFLE when R is removed. When two connected nodes voluntarily quit at the same time, one node should wait until the other node quits successfully (the tie can be broken by the order of IP addresses); if a node crashes while its neighbors are alive, the neighbors can also PUSH or SHUFFLE correctly. However, the above algorithm does not work if node R and its neighbor crash at the same time because both crashed nodes can neither PUSH nor SHUFFLE. We assume that the nodes crash incrementally, one after the other; in the case that a group of connected nodes crash simultaneously, a global restoration algorithm is more appropriate to use.

3.3.2 Degree distribution and P2P search efficiency

Empirical experiments are conducted to evaluate the overlay topologies produced by E-SRA in various settings. These topologies are compared with those produced by HAPA and Gaian algorithms using the same configurations. We measure the fitness of the produced degree distribution to the power-law and then evaluate the search efficiency over these topologies by running Flooding (FL) algorithm and Normalized Flooding (NF) algorithm, which are simple search algorithms used in unstructured P2P networks [107].

By adding and removing nodes dynamically, a set of network topologies are produced with different parameters. At the beginning, a network of $(2k + 1)$ nodes is constructed. Each node connects to all the other $2k$ nodes so that the average degree is $2k$. For all simulations, 1,000 nodes are added first and then we run 150,000 iterations of joining and removing nodes to produce one topology. In each iteration, either a new node joins the network or an existing node is removed. In order to simulate the dynamics of nodes joining and leaving in real-world applications, a node joins with probability $(1 - p)$ and quits with

probability p in one iteration. Thus, a few nodes are quite likely to join or leave the network in a sequence of iterations. The value of p is smaller than $1/2$ to keep the network growing. In a topology produced by E-SRA, nodes with desired degrees may not exist when the network size is small. However, with the growth of the network, there will be sufficient nodes with all possible degrees.

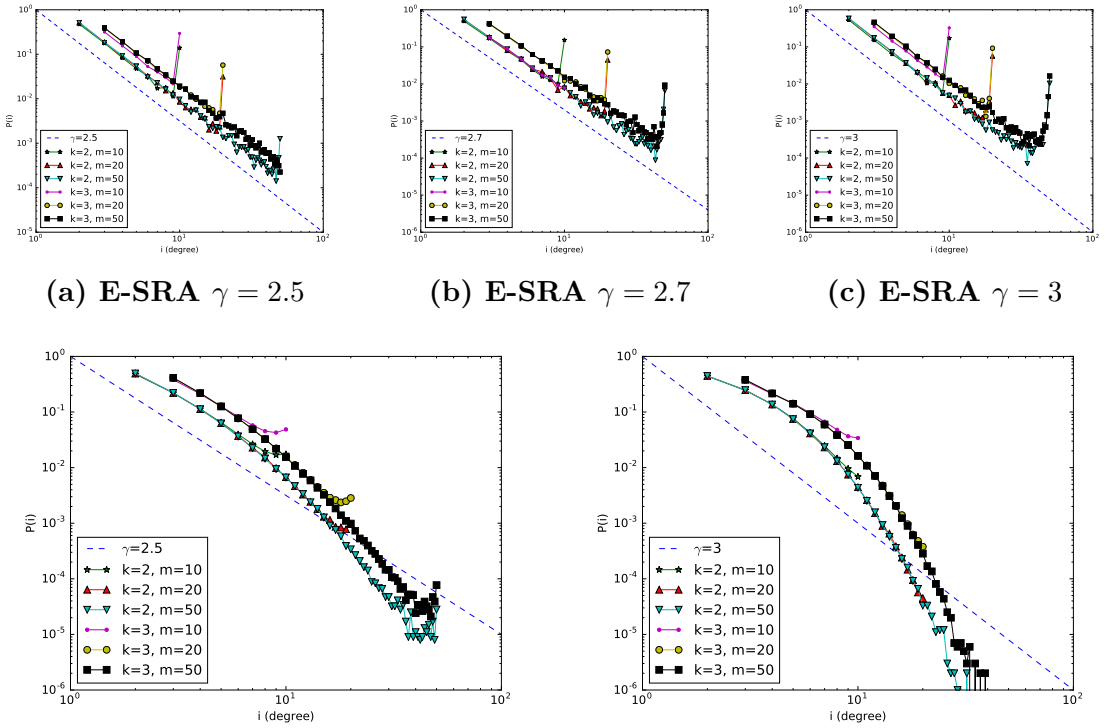
Experiments have been conducted on the topologies produced by HAPA and Gaian algorithms with the same pattern of nodes joining and leaving the network. After a sufficient number of such operations, the degree distribution is calculated based on a "snapshot" of the network topology and is compared with the perfect power-law distribution. Since all three algorithms utilize randomized approaches, we take the average of the degree distribution of 10 randomly produced topologies with the same parameters to study the average cases.

Search algorithms are implemented to test the search efficiency over the produced topologies in different settings. We consider two search algorithms in P2P networks: 1) Flooding (FL), where every node forwards a query to all neighbors until the query hits the target. 2) Normalized Flooding (NF), where every forwarder randomly chooses k (i.e. the minimum degree) neighbors and sends them the query. Time to live (TTL), which is the maximum number of hops a message can traverse, is set up to limit the lifetime of a query in a network. So, a query either reaches its destination or expires due to its TTL. It is assumed that the message sources are uniformly distributed in the network.

It can be observed that regardless of parameter settings E-SRA has produced topologies with the degree distribution perfectly matching the power-law.

Figure 3.13 illustrates the degree distribution of the produced topologies when nodes are removed randomly. In each iteration, either an existing node quits with probability $p = 1/3$ or a new node joins the network with probability $1 - p = 2/3$. The node to be removed is randomly chosen from the network. We apply a total of 1.5×10^5 iterations to produce the final topology. Thus every topology has approximately $(1-2p) \times (1.5 \times 10^5) = 5 \times 10^4$ nodes. In E-SRA, the scaling parameter is set as 2.5, 2.7 and 3 respectively. It can be observed from Figure 3.13 that both Gaian and HAPA algorithms have produced a sufficient number of nodes with low degrees whereas the number of nodes with high degrees is insufficient. Since nodes with high degrees are more likely to connect to the nodes to be removed, their degrees decrease with high probability compared to nodes with low degrees.

As the simulation results suggest, the power-law is approximately preserved in all three



(a) E-SRA $\gamma = 2.5$ **(b) E-SRA $\gamma = 2.7$** **(c) E-SRA $\gamma = 3$**

(d) Gaian **(e) HAPA**

Fig. 3.13. Degree distribution of the topologies where nodes are randomly removed. $n \approx 50000$**,** $p = 1/3$**.**

growth models compared here when the maximum degree is $m = 10$. But, if the maximum degree is $m = 50$, HAPA and Gaian algorithms produce fewer nodes with high degree than needed. This is because the number of nodes with high degrees are of the order of several thousand with $m = 10$, but there are fewer than 100 when $m = 50$. So the final degree distribution is more sensitive to algorithm imprecision and the difference is easier to observe for m=50 than for m=10. For the same reason, E-SRA produces a topology with a small tail at degree 48, 49 when $m = 50$. As the network size grows, the total numbers of nodes at all degrees increase and the tail disappears.

We estimate the scaling parameter of the produced topologies using the MLE method in [98]. In [98], the authors use the Kolmogorov-Smirnov or KS statistic to quantify the difference between the observed degree distribution and the power-law. The smaller is the KS statistic, the closer is the observed degree distribution to the power-law.

In Table 3.5, the degree of removed nodes is denoted as d. When $d \geq 6$, only nodes with degree at least 6 are randomly removed from the topology. As seen in Table 3.5, E-

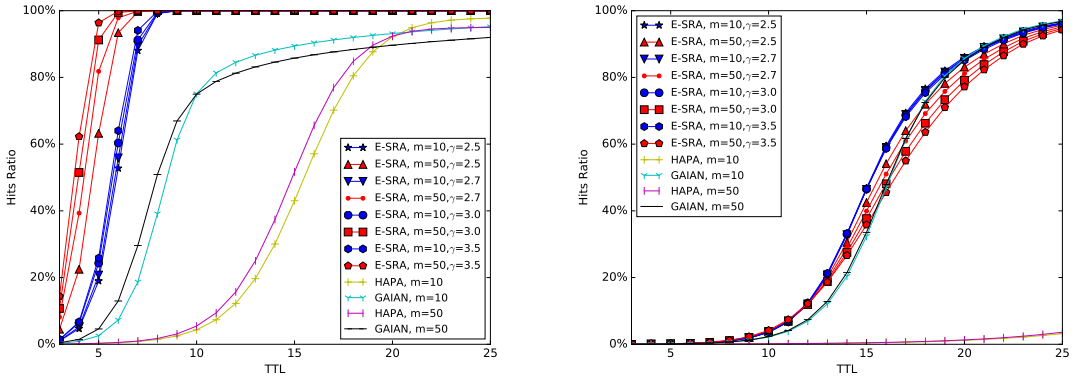
Table 3.5. Results of fitness analysis.

Parameters	Method	E-SRA ($\gamma = 2.5$)	HAPA	Gaian
$m = 20$, $d \geq 6$	γ	2.505169	3.820267	3.574570
	KS statistic	0.002982	0.102532	0.082440
$m = 50$, $d \geq 6$	γ	2.496524	2.672919	2.667210
	KS statistic	0.004415	0.117416	0.078553
$m = 20$, $d < 4$	γ	2.502646	2.283350	2.397095
	KS statistic	0.002524	0.082762	0.023171
$m = 50$, $d < 4$	γ	2.488510	2.269304	2.387625
	KS statistic	0.004728	0.075127	0.010877

SRA produces topologies with a good fit to the scale-free property. The estimated scaling parameters are close to the predefined value $\gamma = 2.5$. In the topologies maintained by HAPA and Gaian algorithms, however, the estimated scaling parameters deviate from the scaling parameter, 3.0 and 2.5, respectively. In all cases, the KS statistics of HAPA and Gaian are larger than those of E-SRA.

We evaluate the search efficiency over the topologies produced by E-SRA, HAPA and Gaian algorithms. We consider two search algorithms commonly used in unstructured P2P networks: 1) Flooding (FL), in which every node forwards a query to all the neighbors until the query hits the target. 2) Normalized Flooding (NF), in which every forwarder randomly chooses k (i.e. the minimum degree) neighbors and sends them the query. Thus, the Normalized Flooding algorithm sets constraints on the number of the messages forwarded by each node. The FL algorithm delivers a query to the destination faster than the NL algorithm in an unstructured P2P network but incurs higher message cost. The Normalized Flooding algorithm sets constraints on the number of the messages forwarded by each node. Both algorithms set time to live (TTL), which is the maximum number of hops a message can traverse, is set up to limit the lifetime of a query in a network. More sophisticated search algorithms in the power-law graphs have been studied in [107].

In order to study the performance of the search algorithms, multiple searching processes are simulated on the same topology. With the same parameters, 10 network topologies are constructed. For each topology, 100 nodes are randomly chosen to broadcast a query, whose the average hit ratio is calculated. Since the destinations of the queries are assumed to be uniformly distributed in the network, the expected hit ratio of a query is proportional to the



(a) FL **(b) NF**

Fig. 3.14. Search Efficiency in networks produced by different approaches and parameters. $n \approx 5 \times 10^4$**,** $p = 1/3$**.**

number of nodes it reaches before the TTL expires. In our experiments, nodes are randomly removed, and the minimum degree is $k = 2$.

Figure 3.14.a shows that a query reaches more than 95% of the network within 7 hops over the topologies produced by E-SRA with $m = 50$ (the red lines); it takes 8 hops for a query to reach 95% nodes in the topologies with $m = 10$ (the blue lines). In the topologies generated by Gaian algorithm, it takes approximately 10 hops for a query to reach 80% but the spreading process becomes much slower after 12 hops. In HAPA, the hit ratio increases very slowly within the first 10 hops and then increases rapidly. Because the topologies produced by E-SRA highly adhere to the scale-free property, the FL algorithm achieves better performance over these topologies.

The search efficiency of the Normalized Flooding algorithm is shown in Figure 3.14.b. One interesting phenomenon is that NF algorithm achieves a higher search efficiency on top of the topologies produced by E-SRA with a small maximum degree m. This is because NF only forwards a query to k neighbors, the nodes with high degrees do not have significant advantages over the nodes with low degrees in forwarding queries. In the first 10 hops, the spreading speed is very slow in the topologies constructed by all three approaches. After 10 hops, the spreading speed becomes much faster. The hit ratio of E-SRA is approximately at least 6% higher than other approaches with the same TTL in the range $[14, 18]$.

It is worth noting that E-SRA can produce the topologies with the user-defined parameters. Therefore, any proper value of the scaling parameter γ could be adopted to produce

the best overlay topology according to simulation results. For example, the FL algorithm achieves the best search efficiency on top of the topologies constructed with $m = 50$ and $\gamma = 3.5$, and the NF algorithm achieves the best search efficiency on top of the topologies constructed with $m = 10$, $\gamma = 2.5$. E-SRA can use these parameters to construct the desired overlay topologies.

3.3.3 Theoretical upper bound on communication overhead

We show that M_{ave}, the average number of the *update message*s sent by a node during its lifetime in the topology, is bounded linearly by the maximum degree m if nodes are randomly removed. Specifically,

$$M_{ave} \leq 3(m - 1 + k)k \tag{3.30}$$

where the minimum degree k is a small constant. We present the proof in detail below.

According to the growth model proposed in previous work [82], when a new node joins the network, it connects to k existing nodes. Among these k nodes, each of the $(1-\sum_{j=k}^{i} f_j)$ nodes needs to send the *update message*s to i neighbors. Thus, if a node joins, the average number of *update message*s sent is,

$$M_{join} = \sum_{i=k}^{m-1} \big(1 - \sum_{j=k}^{i} f_j\big)i \tag{3.31}$$

If a node with degree b quits, then $(b - k)$ PUSH operations and k SHUFFLE operations are required. Due to the SHUFFLE operations, the degrees of $(1 - \sum_{j=k}^{i} f_j)$ nodes decrease from $(i + 1)$ to i. This results in M_{sh} *update message*s.

$$M_{sh} = \sum_{i=k}^{m-1} \big(1 - \sum_{j=k}^{i} f_j\big)i \tag{3.32}$$

The $(b-k)$ PUSH operations change the degrees of $(b-k)$ nodes, which leads to a total of $(\sum_{i=k}^{b-1} i)$ *update message*s. If nodes are randomly removed from the scale-free topology,

the average number of *update message*s caused by the PUSH operations is,

$$M_{push} = \sum_{b=k+1}^{m} \big(f_b \sum_{i=k}^{b-1} i\big) \tag{3.33}$$

Since $\sum_{j=k}^{m} f_j = 1$, we have,

$$\sum_{i=k}^{m-1} \big(1 - \sum_{j=k}^{i} f_j\big) i = \sum_{i=k}^{m-1} \big(\sum_{j=i+1}^{m} f_j\big) i = \sum_{b=k+1}^{m} \Big(f_b \sum_{i=k}^{b-1} i\Big) \tag{3.34}$$

Thus,

$$M_{push} = M_{sh} = M_{join} = \sum_{b=k+1}^{m} \big(f_b \sum_{i=k}^{b-1} i\big) \tag{3.35}$$

Using Eq. 3.18, the rightmost part of the above equation can be simplified as,

$$\begin{aligned}
&\textstyle\sum_{b=k+1}^{m} \Big(f_b \sum_{i=k}^{b-1} i\Big) \\
&= \textstyle\sum_{b=k+1}^{m-1} \Big(f_b \sum_{i=k}^{b-1} i\Big) + (1 - \sum_{b=k}^{m-1} f_b) \sum_{i=k}^{m-1} i \\
&= \textstyle\sum_{i=k}^{m-1} i + \sum_{b=k}^{m-1} \Big[f_b \big(\sum_{i=k}^{b-1} i - \sum_{i=k}^{m-1} i\big)\Big] \\
&= \textstyle\sum_{i=k}^{m-1} i - \sum_{b=k}^{m-1} \Big[f_b \big(\sum_{i=b}^{m-1} i\big)\Big]
\end{aligned} \tag{3.36}$$

Using the definition of f_i in Eq. 3.17, the rightmost part of the above equation can be rewritten as,

$$\begin{aligned}
&\textstyle\sum_{i=k}^{m-1} i - \sum_{b=k}^{m-1} \Big[\frac{m-2k}{b^\gamma \sum_{j=k}^{m-1} \frac{m-j}{j^\gamma}} \big(\sum_{i=b}^{m-1} i\big)\Big] \\
&= \frac{(m-1+k)(m-k)}{2} - \frac{m-2k}{2} \frac{\sum_{b=k}^{m-1} \frac{(m-1+b)(m-b)}{b^\gamma}}{\sum_{j=k}^{m-1} \frac{m-j}{j^\gamma}} \\
&\leq \frac{(m-1+k)(m-k)}{2} - \frac{m-2k}{2} \frac{(m-1+k) \sum_{b=k}^{m-1} \frac{(m-b)}{b^\gamma}}{\sum_{j=k}^{m-1} \frac{m-j}{j^\gamma}} \\
&= \frac{(m-1+k)(m-k)}{2} - \frac{(m-2k)(m-1+k)}{2} \\
&= \frac{(m-1+k)k}{2}
\end{aligned} \tag{3.37}$$

Consider a network with n nodes, where N nodes have been added to the network ($N \gg n$). A total of $(N - n)$ nodes have been removed from the network. Every removal results in $(M_{push} + M_{sh})$ *update message*s and every joining results in M_{join} *update message*s. If nodes are randomly added and removed over a long period, the average number of *update*

*message*s sent by a single node is,

$$\begin{aligned} M_{ave} &= \lim_{N\to\infty} \frac{NM_{join}+(N-n)(M_{push}+M_{sh})}{N} \\ &= 6\sum_{b=k+1}^{m}\left(f_b\sum_{i=k}^{b-1} i\right) \leq 3(m-1+k)k \end{aligned} \tag{3.38}$$

Since the minimum degree k is a constant value, the average number of P2P *update message*s is linear to the hard degree cut-off m. Specifically, when $k = 2$, the average number of *update message*s sent by a single node is at most $6(m + 1)$.

3.4 Conclusions

In this chapter, we explore the news event dataset, GDELT, to understand the structure of the online media and the emergent propagation patterns of the news events among online news sites around the world. A novel framework is proposed to infer the influence and selectivity of online media sites on a variety of topics. Based on the stochastic gradient descent algorithm, we propose an efficient inference algorithm to derive such influence and selectivity of the online media sites. We successfully parallelized this algorithm for distributed memory machines and tested this parallelization on the RPI Advanced Multiprocessing Optimized System (AMOS) achieving orders of magnitude speedup. Hence, the framework can be applied to the large dataset GDELT which includes thousands of news sites and millions of news events.

In this section, we use machining learning models to predict the number of news sites reporting one specific event. By extracting many features of a news event at its early stage, the model achieves a decent accuracy on predicting the final virality of the news event. This approach also reveals the importance of the extracted features on the eventual popularity of the news events, which would shed light into the rapid growth of the reports on some critical news events in online media.

We exploit the latent community structure in the global news network to improve the prediction of the viral cascades of news about events. The cascades which have early adopters in different communities have advantages in disseminating the contagion to these communities in parallel, and therefore are more likely to result in the viral infections within a limited period. Our model captures such property by inferring the community structure using the response times of nodes rather than using the explicit network topology because

the references to propagation sources are usually missing in the real datasets.

Besides the prediction of viral information cascades, we also propose a growth model to construct the scale-free overlay topology which facilitates the information cascades in unstructured peer-to-peer (P2P) networks. Previous growth models can maintain the limited scale-free topology when nodes only join but do not leave the network; the case of nodes leaving the network is not included in the solution. Thus, the full dynamic of node participation, inherent in P2P networks, is not considered in these models. In order to handle both nodes joining and leaving the network, we propose a robust growth model E-SRA, which is capable of producing the perfect limited scale-free overlay topology with user-defined scaling parameter and hard cut-off. Scalability of our approach is ensured since no global information is required to add or remove a node. E-SRA is also tolerant to individual node failure caused by errors or attacks. Simulations have shown that E-SRA outperforms other growth models by producing topologies with high adherence to the desired scale-free property. Search algorithms, including flooding and normalized flooding, achieve higher efficiency over the topologies produced by E-SRA.

CHAPTER 4
POLITICAL POLARIZATION IN THE LEGISLATIVE BRANCHES

4.1 Introduction

Conflict and consensus play a vital role in the functioning of a social system. In the context of political competition, they manifest themselves as the polarization of opinions and collaboration to reach consensus on shared national interests [108]. Polarization arises from the politicians' need to represent the opinions of their voters while collaboration is required to balance the interests of many groups.

Human behavior is profoundly affected by the propagation of opinion and influence through social connections of individuals. Numerous previous publications have focused on the role of social conformity [109]–[111] in polarization. Among these publications, many hypotheses have been proposed to explain the observed emergence of increased polarization, including social homophily [20], selective exposure [19], social bots [112], echo chambers [18], [113], propagation of low-quality information or fake news [16], [17], as well as the effect of viral news [14] and social media [114]. While these models study different aspects of polarization of political views, they share some common assumptions about human social behavior [21], including the following: (i) individuals iteratively update their views to reach consensus with their neighbors in a social network; (ii) the tolerance of conflicting views is limited in social context, so frequent active disagreements usually break social ties. These assumptions indicate that the loyalty to one's group usually leads to the conformity with views of the group's majority [22], and such conformity tightens social ties within the group. Therefore, in our model, we allow the current polarization level to influence its future growth.

The polarization of political opinions among members of the U.S. legislative chambers measured by their voting records is higher today than it was thirty years ago. In the Section 4.3, we analyze millions of roll-call votes cast in the U.S. Congress [115] over the past six decades. While the behavior of individuals in a social system is highly unpredictable [116],

Portions of this chapter are submitted as:

X. Lu, J. Gao, and B. K. Szymanski, "The evolution of polarization in the legislative branch of government," submitted for publication

[117], the collective motion of the social system shows a continuous evolution pattern. In our model, we assume that, at the beginning of each Congress, the system starts from a level of polarization determined by election. Thus, the corresponding dynamical system has different starting points at the beginning of each term. However, as the members collaborate and compete with each other during each Congress term, the dynamical system starts moving towards the polarization level of convergence. A hidden parameter, called polarization utility, is critical for the convergence of polarization level at the election of each Congress. From the roll-call votes cast in the U.S. Congress [115] over the past six decades, we derive the polarization utility of our model by fitting our model to the real data.

According to the analysis on the voting dataset, we witness a growth of the polarization over the recent decades. The question arises what the causes of this increase are. To answer this question, we start by observing that a politician needs to be elected to become legislators and repeatedly reelected to continue in this role. Thus, the polarization utility for them lies in its ability to bring votes. This can be accomplished either directly, by representing voters' opinions, or by gaining resources for election campaign funding. In the case of direct support, the polarization of voters has been rising in recent decades for such reasons as the echo chambers effect [18], [113] and the growth of new, often strongly biased social media [114] or spread of misinformation [16], [17]. Voters increasing polarization raises the polarization utility, which is indirectly reflected as the phenomenon that legislators align their voting with positions of their electorates. The indirect support is gaining in importance because of escalating costs of political campaigns fueled by the growing numbers of active advertising channels and rising costs of advertising [118].

Section 4.4 presents two most significant jumps in polarization utility, a hidden parameter of our model, resulting from the election of Congress members. The first jump of 0.24 happened in 1960, so it coincides with the start of the civil right movement, increasing the U.S. involvement in the Vietnam War, and generational changes in politics. In [119], the author observes that such "takeoff situations," significantly increase polarization in the network structures of political connections. The second jump happened in the year 2010 when the Supreme Court approved Super PACs which are allowed to collect unlimited contributions from many sources and to advocate for or against political candidates.

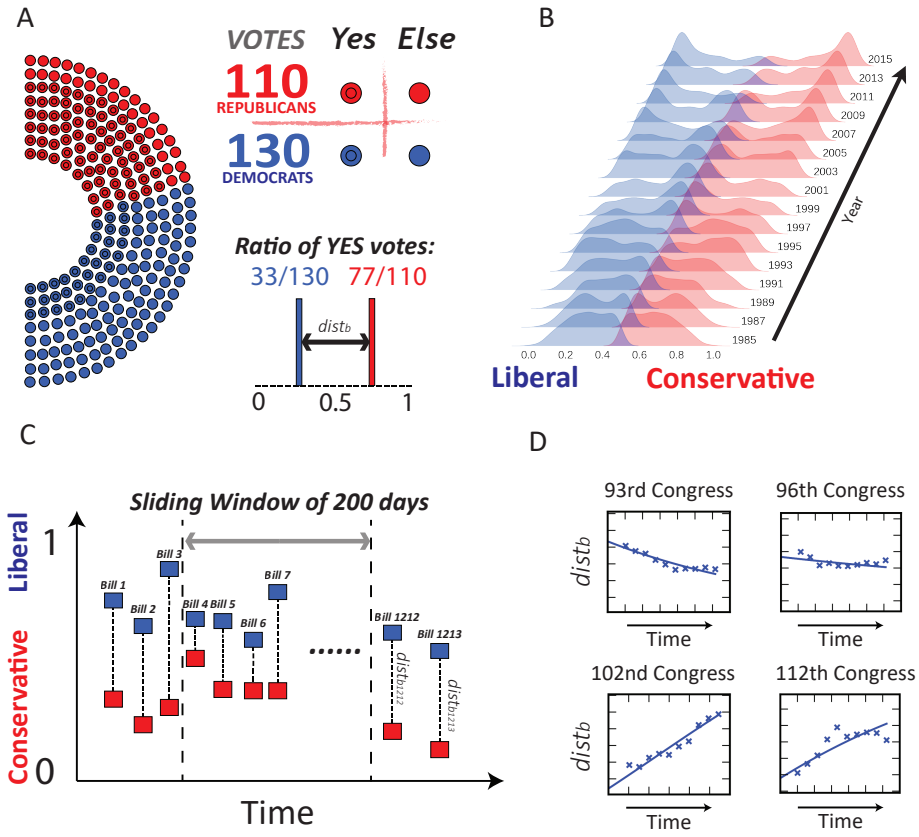


Fig. 4.1. The illustration of the data collections and processing workflow. (A) For each bill voted in the Congress, we measure the mean absolute distance, $dist_b$, between the votes of the Democratic and Republican Parties cast for the bill b; (B) The distribution of the political polarization measured using the roll-call votes within each Congress. Labels at the right corner of each sub-plot identify the first year of each Congress. In the 1980s and 1990s, the polarization levels are generally smaller than the levels in the 2000's and 2010's. After 2001, the two peaks of the voting results at the opposite ends of the political spectrum start to emerge, indicating the growth of the number of bills on which two parties strongly disagree; (C) Regardless of the content of bills, we compute the average distance of bill votes between two parties within each Congress in a sliding window of 200 days (Eq. [4.2]); (D) The political polarization levels at ten evenly-distributed sampled time points exhibit an evolution of polarization patterns from one type of behavior to another: the polarization level decreases in the 93rd Congress as time from the replacement of members increases, while polarization remains at a relative stable level in the 96^{th} Congress, and grows in the Congresses with sessions numbered from 102 to 112.

Table 4.1. The statistics of the roll-call votes from the 85th to the 114th U.S. Congress.

Party	#Members	#Bills	#Votes	#Votes per member	#Votes per year
Democratic	1498	31,879	7,368,921	4919.17	122815.35
Republican	1395	31,879	6,275,886	4498.84	104598.1

4.2 Quantifying polarization in the legislative branch

We analyze millions of roll-call votes cast in the U.S. Congress [115] over the past six decades to identify evolution of political polarization patterns. The statistics of the roll-call votes in Congresses with sessions numbered from 85 to 114 are shown in Tab. 4.1. The bills which received less than 30 votes are not included in the statistics. This dataset contains approximately 7 million votes in both the Senate and the House of Representatives from a total of 1498 and 1395 legislators from Democratic and Republican Parties, respectively. We adopt the well-known Rice index [120] to measure party dissimilarity in legislative voting. The Rice index is defined as the mean absolute distance between the Yes-ratios of Democratic and Republican Parties on the b^{th} bill

$$dist_b = \left| E_{\{1 \le i \le N_{Rep}\}} \mathrm{Rep}_{ib} - E_{\{1 \le j \le N_{Dem}\}} \mathrm{Dem}_{jb} \right| \tag{4.1}$$

where N_{Rep} and N_{Dem} denote the numbers of legislators from the Republican and Democratic Parties participating in the vote, while Rep_{ib} and Demo_{jb} are the votes cast by Republican i and Democrat j for bill b, respectively. E represents the corresponding average over all legislators in each party. The result of a vote is coded as 1 for Yes and 0 otherwise because the bills pass by majority in both the Senate and the House of Representatives and therefore abstaining is effectively equivalent to opposing bill passage. This procedure is illustrated in Fig. 4.1A.

Regardless of the content of bills, we compute the average distance of bill votes between two parties over every 199 day intervals. Formally, the polarization level at the t^{th}h day of the k^{th} Congress is quantified as

$$x_k(t) = E_{\{b:|t_b - t|<100\}} \left[dist_b \right] \tag{4.2}$$

where $\{b : |t_b - t| < 100\}$ is the set of bills voted within 199 days centered at the th day of the

i^{th} Congress and i starts at the 100^{th} day of each Congress and ends 99 days before the last day of this Congress. Hence, each measurement includes exactly 200 days of voting. This step is illustrated by Fig. 4.1C. The averaging reduces the noise of the raw data because the topics of the legislative bills may differ day-by-day in each Congress and there were several periods of times in the past six decades during which very few bills were voted on by the U.S. Congresses. Compared to the approaches [119], [121], [122] defining the conflicting level between individuals, the well-know Rice index defined in Eq. [4.1], reflects the general trend of behavioral partisanship while preserving its simplicity by quantifying polarization at party level. More importantly, the Rice index enables us to develop a dynamical equation, Eq. [4.3], which captures the macroscopic behavior of the evolution of political polarization, regardless of the complex interactions between individual legislators considered in [121].

For every Congress and each bill, we measure the distance $dist_b$ between two parties using Eq. [4.1]. As $dist_b \in [0, 1]$, the political preference for bill b is defined as $D_b = 0.5 - \frac{dist_b}{2}$ for the Democratic Party and $R_b = 0.5 + \frac{dist_b}{2}$ for the Republican Party. Then, using the kernel density estimation (KDE) [123], we evaluate the distribution of distances D_b and R_b, of Democratic and Republican Parties, respectively, from the center of the polarization range. Fig. 4.1B shows the distribution of these distances which represent positions of the two parties regarding the bills voted in each Congress.

4.3 Dynamical model of political polarization

We assume a two-party political system, such as exemplified by the United Kingdom, but applicable also to the U.S. and other countries with a multi-party system dominated by two major parties. We also assume a social system in which polarization and collaboration can convert into each other but preserve their sum at 1. A simple model of the dynamics of such conversion is given in [124] as

$$\frac{dx}{dt} = yP_{yx}(x, u_x) - xP_{xy}(x, u_x) \tag{4.3}$$

where $P_{yx}(x, u_x)$ is the probability of collaboration converting to polarization per unit of time. For symmetry, $P_{xy}(x, u_x)$ represents the probability of polarization converting to collaboration per unit of time. Finally, $x \in [0, 1]$ is the current polarization level as measured by the real legislative votes, u_x is the polarization utility parameter, and $y = 1 - x$ and

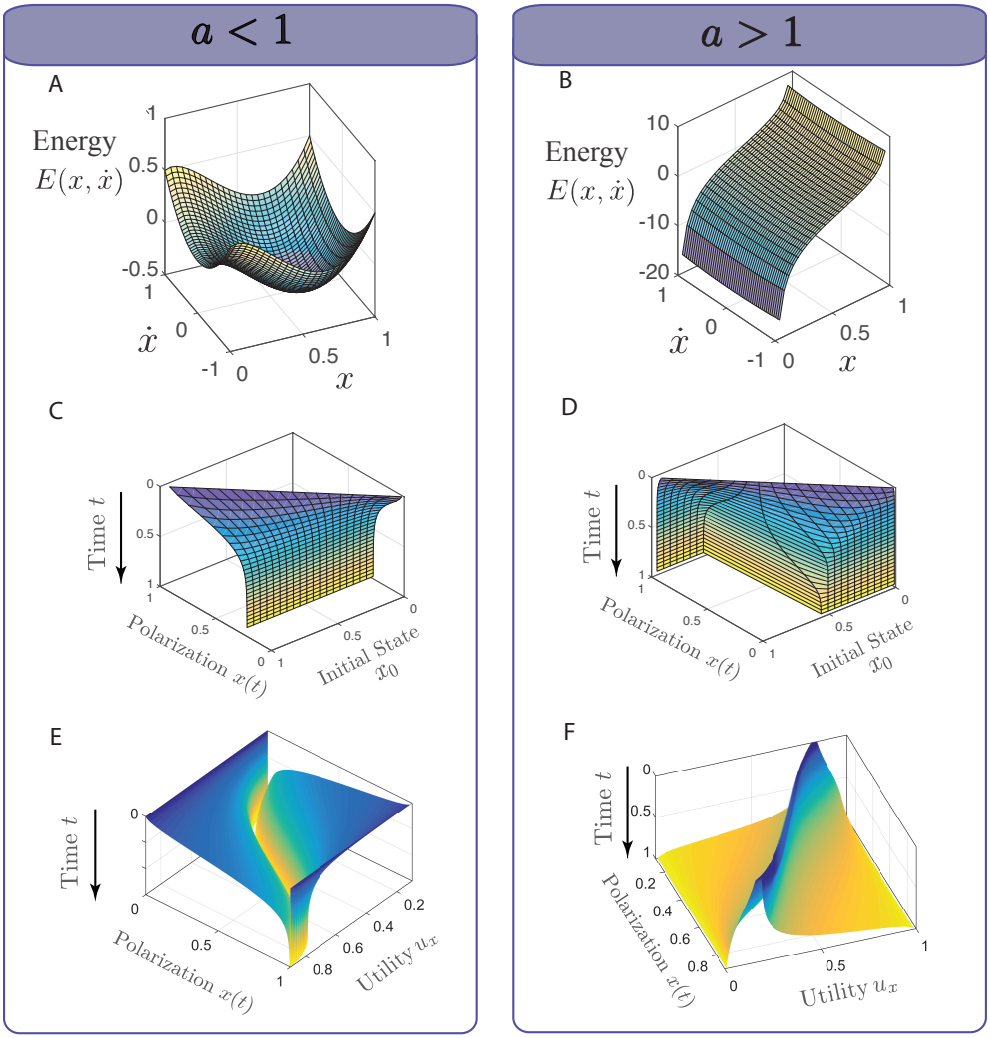


Fig. 4.2. The convergence of the political polarization model in relation to the polarization utility u_x and initial state x_0. The time t is normalized in the plots to the range from 0 to 1. (A) The total energy of the dynamical system in relation to the polarization x and its first order derivative $\dot{x}$ for $a < 1$ (Eq. [4.6]); (B) The total energy of the dynamical system in relation to the polarization x and its first order derivative $\dot{x}$ for $a > 1$ (Eq. [4.6]); (C) For $a < 1$ the dynamical system always converges to certain polarization level ($a = 0.6$); (D) For $a > 1$ the dynamical system either reaches complete consensus or complete polarization depending on the initial state x_0 ($a = 2.5$). (E) When $a < 1$, the equilibrium points of the dynamical system are stable as the system gets trapped at these equilibrium points as the time approaches infinity; (F) When $a > 1$, the initial states (on the top of the hills) are the tipping points causing the system to converge either to 0 (full polarization) or 1 (full consensus) from its initial state x_0.

$u_y = 1 - u_x$ are complementary to x and u_x, representing the current collaboration level and the collaboration utility ($u_x + u_y = 1$). Following [124], we assume that P_{yx} has the following simple form

$$P_{yx}(x, u_x) = cx^a u_x \tag{4.4}$$

which is supported by the normative social influence theory [22] that postulates that the current polarization level x influences the probability of conversion. To preserve symmetry under the conversion from y to x or vice versa, we define P_{xy} as

$$P_{xy}(x, u_x) = P_{yx}(y, u_y) = cy^a u_y = c(1-x)^a(1-u_x) \tag{4.5}$$

Similar nonlinear gain-loss equations for the state dynamics have been successfully applied to model various types of polarization, ranging from religious affiliation [124], to language choice [125] and political affiliation [126]. The nonlinear dynamics defined here captures the conversion between polarization and collaboration during different periods. The parameter u_x is independent of the polarization level x which varies as the time changes. It reflects internal hidden polarization level in the legislative branch of government, which is not reflected from the vote results at that moment. The term x^a captures the effect of the current polarization level on the evolution. The speed of evolution is defined by the parameter c in our model.

The total energy [127] in this dynamical system is governed by the following equation.

$$E(x, \dot{x}) = \frac{1}{2}\dot{x}^2 - cxy\Big(\frac{u_x}{x^{1-a}} - \frac{u_y}{y^{1-a}}\Big) \tag{4.6}$$

which is constant on the solution curves or trajectories of this system, i.e. $E(x, \dot{x}) = C$ for some constant C. The total energy function $E(x, \dot{x})$ in relation to the current polarization x and its first order derivative $\dot{x}$ is shown on the Fig. 4.2A for $a < 1$ and Fig. 4.2B for $a > 1$.

Given the model parameters a and u_x, the equilibrium points x^* of the dynamical system defined by Eq. [4.3] can be derived as

$$x^*(u_x, a) \approx \frac{1}{1 + e^{2\frac{1-2u_x}{1-a}}} \tag{4.7}$$

The approximation uses the Taylor expansion of $\dot{x} = 0$. Its detailed derivation is presented in the supporting material. The theoretical expression of equilibrium points closely matches the results found by numerical simulations.

The trajectories of the nonlinear dynamical evolution are shown in Fig. 4.2C-F where the x-axis represents time t and the y-axis represents the polarization. Each trajectory curve starts from its initial polarization level x_0 which is represented by the z-axis. We show the surface of these convergence process for $a < 1$ and the initial polarization in full range from $x_0 = 0$ to $x_0 = 1$, in Fig. 4.2E. Fig. 4.2D contains similar visualization results with $a > 1$ and initial polarization levels x_0 at the tipping points, i.e. the points at the edges in dark blue of Fig. 4.2F. The equilibrium points of the dynamical model are stable when $a < 1$. In such case, the dynamical system of Eq. [4.3] always converges to the $x^* \in [0, 1]$ after the sufficient period of evolution; Fig. 4.2C and 4.2E show example with $a = 0.6$. In contrast, when $a > 1$, the final state of the dynamical system depends on the initial state and the position of the tipping point; Fig. 4.2D and 4.2F show example with $a = 2.5$. The dynamical system will eventually converge to either $x = 0$ or $x = 1$.

In the context of political polarization, the case with $a < 1$ corresponds to a healthy political system which maintains the polarization level within a certain range. However, the case with $a > 1$, corresponds to a system which switches easily between fully polarized equilibrium point and complete consensus convergence point. In the U.S. political system, a change of the system initial state happens periodically every two years as a result of election or reelection of the legislators. The system near the tipping point at the end of one Congress is vulnerable to extreme state switching even under a small change of the initial polarization for the next Congress.

We simulate the non-linear dynamics defined by Eq. [4.3] until the polarization level x converges. When $a < 1$, the equilibrium point is exactly the final polarization level of convergence. When $a > 1$ the only equilibrium points are $x = 1$ that corresponds to the full polarization, and $x = 0$ that represents the opposite case of full consensus. Moreover some of the initial states are unstable; they are the tipping points from which the dynamical system nondeterministically converges to either of the two stable equilibrium points. We grid-search for them with different values of u_x. In each iteration, we increase the value of x by a small Δx. If the final convergence states have changed from one state to another due to the increment of Δx, then the current x is identified as the tipping point.

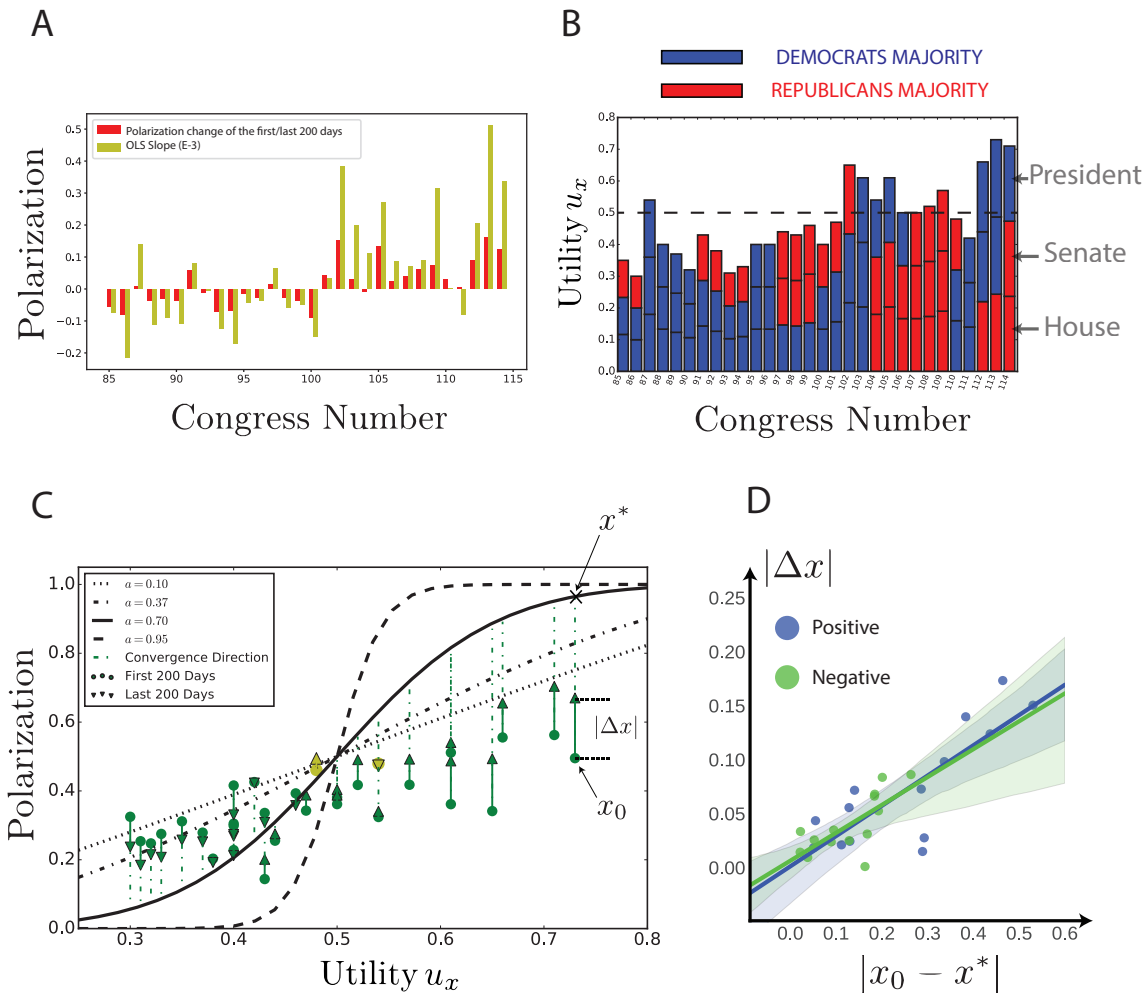


Fig. 4.3. The evolution of the political polarization in the U.S. Congress with subsequent sessions numbered from 85 to 114.

4.4 Evolution of political polarization patterns

We fit the political polarization model defined by Eq. 4.3 to the data points $x_i(t)$ which represent ten evenly-distributed in time sampled polarization levels at $t_i = t_{i,1}, t_{i,2}, \ldots, t_{i,10}$ of the th Congress. The system is assumed to have the universal a and c values at all times, but each Congress i has its own parameter u_x defined as the polarization utility, and the particular initial state, $x_{i,0}$, which is defined by the member replacements caused by the most recent election.

The inference procedure estimates the values of universal parameters at $a = 0.7$, $c = 0.37$ while the set of values for u_x, x_0 is shown in Fig. 4.3B-C. In Fig. 4.3B, we illustrate

the estimated value $u_{i,x}$ of the i^{th} Congress. In general, this value increases as the Congress session numbers grow from 85 to 114. The black solid curve in Fig. 4.3C shows the equilibrium points for different u_x values with global parameter a estimated from the voting data. To illustrate the impact of this parameter on values of the equilibrium points, the dotted lines represent these values for $a = 0.1$, $a = 0.37$ and $a = 0.95$, respectively. This figure also illustrates that most of the Congresses last too short to allow the equilibrium point to be reached before the election or reelection of members defines new starting point of the system for the next Congress.

Fig. 4.3(A) shows the change of polarization within each U.S. Congress based on data and the optimal slope parameters estimated by the Ordinary Least Squares (OLS) linear regression given the ten evenly-distributed in time points of polarization sampling for each Congress. The polarization level is likely to decrease within each Congress after the member replacement in the 1970's and 1980's, however, after the 101^{th} Congress, which started in 1989, the polarization level is likely to increase instead. As illustrated in Fig. 4.3(B), the estimated values of the polarization utility u_x are generally increasing, while the periods of sharp growth are often associated with the change of majorities in the Senate and the House of Representatives. In Fig. 4.3(C), when the initial polarization level (green cycles) is smaller than the stable polarization level predicted by our model (solid black curve), we observe an increase of polarization within one Congress. The direction of such change in 28 out of all 30 Congresses are explained by the model (green arrows), while the two Congresses in disagreement with prediction have minimal variations of the polarization level (yellow arrow markers) which indicates weakly formed polarization direction. In general, when the initial polarization level x_0 of a Congress is farther away from the corresponding equilibrium point, the absolute change of polarization during the two-year term, i.e. $|\Delta x|$, of this Congress is usually higher than of Congresses in which the initial polarization levels are closer to the equilibrium points. This result is shown in Fig. 4.3(D) where the color of the scatter points indicates the sign of Δx.

Since the estimate $a = 0.7$ is smaller than 1, the dynamical system has a unique equilibrium point for any given u_x. As shown in Fig. 4.3C, when the polarization level x is larger than the stable equilibrium point x^*, it decreases to approach this point. This explains why the polarization level decreases in most of the Congresses in the 1970's and 1980's. After the member replacement caused by the election, the initial polarization levels

x_0's during this period were usually higher than the corresponding stable equilibrium points of the dynamical system, see Fig. 4.3C. Therefore, the polarization level decreases within the two-year term Congress. In contrast, when the equilibrium point x^* is larger than the initial polarization x_0, the polarization x increases over time to approach the equilibrium point. This explains why the polarization level increases in most Congresses after the 1980's. Hence, the observed polarization patterns are fully recreated by the dynamics of our model.

Table 4.2. The estimated values of the polarization utility (P-utility) u_x in each Congress.

Midterm Election Congresses				**Presidential Election Congresses**			
Number	**P-utility**	**Change**	**Percentage**	**Number**	**P-utility**	**Change**	**Percentage**
86^{th}	0.3	-0.05	-14.3%	$\mathbf{87^{th,*}}$	**0.54**	0.24	80.0%
88^{th}	0.4	-0.14	-25.9%	89^{th}	0.37	-0.03	-7.5%
90^{th}	0.32	-0.05	-13.5%	91^{st}	0.43	0.11	34.4%
92^{nd}	0.38	-0.05	-11.6%	$93rd$	0.31	-0.07	-18.4%
94^{th}	0.33	0.02	6.5%	$95th$	0.4	0.07	21.2%
95^{th}	0.4	0.00	0.00%	$97th$	0.44	0.04	10.0%
98^{th}	0.43	-0.01	-2.3%	99^{th}	0.46	0.03	7.0%
100^{th}	0.4	-0.06	-13.0%	101^{st}	0.47	0.07	17.5%
102^{nd}	0.65	0.18	38.3%	$103^{rd,*}$	**0.61**	-0.04	-6.2%
104^{th}	0.54	-0.07	-11.5%	$105^{th,*}$	**0.61**	0.07	13.0%
106^{th}	0.5	-0.11	-18.0%	$107^{th,*}$	**0.5**	0.00	0.0%
108^{th}	0.52	0.02	4.0%	$109^{th,*}$	**0.57**	0.05	9.6%
110^{th}	0.48	-0.09	-15.8%	111^{th}	0.42	-0.06	-12.5%
$112^{th,*}$	**0.66**	0.24	57.1%	$113^{th,*}$	**0.73**	0.07	10.6%
114^{th}	0.71	-0.02	-2.74%				
86th-100th	0.370	-0.043	-9.3%		0.421	0.056	18.1%
101st-114th	0.580	0.021	7.3%		0.559	0.023	4.6%
All	0.468	-0.013	-1.5%		0.481	0.039	11.3%
	4 positive, 10 negative, 1 *				**9 positive, 4 negative, 6 ***		

* denotes growth over 50%, marked also by bold font in the P-utility (polarization utility) column

It is worth noting that the initial polarization levels of the Congresses in the 1990's are actually not significantly higher than those observed in the previous Congresses. However, the polarization utility u_x has become larger and in the later decades exceeded 0.5 as Fig. 4.3B and Table 4.2 indicate. Consequently, the polarization levels at the end of Congresses have significantly increased. This explains the rapid growth of polarization in the later Congresses. The sudden growth of polarization utility in the 101^{st} and 102^{nd} Congresses (1989-1993), revealed by our model is in agreement with [121] which describes a dramatic change of polarization that started during the Clinton's term (1993-1994), and solidified during the

104^{th} Congress (1995-1996). Thus, the growth of polarization utility came right before the growth of polarization because legislators need time to adjust voting to the increased polarization which has been reflected by the polarization utility. As seen in Fig. 4.3D, the absolute change of polarization $|\Delta x|$ grows as the distance between the initial state x_0 and equilibrium point x^* increases. If the initial polarization level of a Congress is far away from the its final point of convergence, then the rapid change of polarization would be expected within the two-year term. The direction of such change in 28 out of all 30 Congresses are explained by the model, while the two Congresses in disagreement with the model prediction have very small variations of polarization level as shown by the yellow markers in Fig. 4.3C.

As shown in Table 4.2, the 14 Congresses with the presidential election held in the preceding year increase u_x on average by 11.3% since the corresponding previous Congress while the 15 Congresses with midterm elections (as the President passes half of his term at the time of the election), decreased the polarization utility u_x by -1.5% on average. Moreover, in the first three decades, the average polarization utility grew slowly by 0.056, so 18.1% on average in Presidential election Congresses. All this growth was gained in four Presidential elections in which the newly elected President and his predecessor belonged to different parties; each of these elections contributed growth of 0.115. so 36.4%, on average. In contrast, the polarization utility decreased by -0.043 or -9.3% in midterm election Congresses, raising only 14.3% over 30 years. In the latest three decades, the polarization utility grew in both types of Congresses with similar average rates, of 0.023 or 4.6% for midterm election Congresses, and 0.21 or 7.3% for Presidential election Congresses. From the 100^{th} Congress to 114^{th} Congress the polarization grew 77.5%, so five times higher than in the earlier period. Finally 6 of 14 Presidential election Congresses started with polarization at least 50% while only one of 15 midterm election congresses achieved such high polarization.

In summary, our model explains the observed polarization patterns. In the 1970's and 1980's, the initial polarization x_0 is generally larger than the stable polarization x^*, so we observe a decrease of polarization within each of two-year term Congress. In other words, the legislators gradually agree more and more with the members of the opposite party than initially, while they held more conflicting views at the very beginning of each Congress session. After the 101^{st} Congress (1989-1991), the initial polarization x_0 is generally smaller than the stable polarization level x^*. Therefore, during the corresponding Congresses, the polarization x increases to approach the corresponding equilibrium point defined by the given

u_x. This trend matches the transition observed during the 103^{rd} and 104^{th} Congresses [121], when the moderate members of each party joined their majority-party coalitions, leaving the middle ground deserted.

The dynamical model sheds some light on the causes of increased polarization in recent decades. As seen in Fig 4.3C, when u_x is small, the resulting stable polarization level decreases as the value of a increases, in response to the decreased probability of conversion from collaboration to polarization (cf.Eq. [4.3]). In this case, the dynamical system is less sensitive to the current polarization level x because the term x^a in Eq. [4.3] becomes smaller as a grows. Therefore, when u_x is small, a large value of a decreases the polarization level. However, when the value of a exceeds 1, some initial states become tipping points, from which the system evolves undeterministically towards one of the two possible extreme equilibrium points, one of which, $x = 1$, corresponds to the full polarization, while the other, $x = 0$, represents the full consensus. Below the tipping point, the system converges to one of these two points, and above, it evolves towards the other. In the U.S. political system, a change of the initial system state happens periodically every two years when the legislators are elected or reelected. The system in the neighborhood of a tipping point is vulnerable to even small change in initial polarization that may switch the system convergemce point from one extreme equilibrium point to another. Such abrupt and radical equilibrium point switching is absent when a is smaller than 1.

According to our model, we witness a growth of the polarization utility, u_x, over the recent decades. The question arises what are the causes of this increase. To answer this question, we start by observing that a politician needs to be elected to become legislators and repeatadly reelected to continue in this role. Thus, the polarization utility for them lies in its ability to bring votes. This can be accomplished either directly, by representing voters' opinions, or by gaining resources for election campaign funding. In the case of direct support, the polarization of voters has been rising in recent decades for such reasons as the echo chambers effect [18], [113] and the growth of new, often strongly biased social media [114] or spread of misinformation [16], [17]. Voters increasing polarization raises the polarization utility, which is indirectly reflected as the phenomenon that legislators align their voting with positions of their electorates. The indirect support is gaining in importance because of escalating costs of political campaigns fueled by the growing numbers of effective advertising channels and raising costs of advertising [118].

To corroborate this conclusion, we identified two largest jumps in polarization utility resulting from election of Congress members (see Table 4.2). The first jump of 0.24 happened in 1960 so it coincides with the start of civil right movement, increasing the U.S. involvement in the Vietnam War, and generational changes in politics. In [119], the author observe that such "takeoff situations," significantly increase polarization in the network structures of political connections. The second jump happened in year 2010, when the Supreme Court approved Super PACs which are allowed to collect unlimited contributions from many sources and to advocate for or against political candidates. Taming the causes of increased polarization is difficult. For example, requireing biased social media to provide time to advovates of the opposing opinions was shown to be counter productive [128].

The model introduced here implies that high polarization of voters makes the polarization utility higher to legislators, resulting in higher polarization in the legislative chambers of government. The question arises under what conditions the polarized politicians may in turn influence their electorates to become polarized even more. Finding these conditions is important because should such feedback loop arise, it might destabilize democracy. In our future research, we will attempt to address this question by developing a quantitative model of polarization dynamics of voters, to large extent shaped by the economic factors and public opinions [129].

4.5 Conclusions

In this section, we present a dynamical system modeling the evolution of polarization in the legislative branches dominated by two parties. In such a system, polarization and collaboration can convert into each other, but they maintain their sum constant. The convergence level of polarization is determined by a model parameter called polarization utility in analogy to the role of gravitational utility in physics.

Studying polarization in the U.S. Congress, we assume the nonlinear dynamics fully govern the evolution of polarization. However, every two years, the dynamical system moves to a new state determined by the election of members of the next Congress for which this state then becomes the initial state. In other words, the replacement of members in each Congress caused by election sets the new initial polarization level. Up to the end of the 1980s, the polarization level within each Congress tended to decrease. Since then, however, the polarization has been likely to grow within the two-year term of each Congress. This

phenomenon is represented in our model by the change of the polarization utility, since it determines the polarization level to which the system converges, regardless of its initial state. The non-linear dynamics successfully predict the direction of polarization change in 28 out of 30 U.S. Congresses for the past six decades. The two Congresses that in disagreement with the model predictions have minimal variations of polarization, with a weakly-defined polarization direction, making the prediction error small.

Moreover, our result suggests that farther away is the initial polarization caused by member replacement from the corresponding equilibrium defined by our model, faster the polarization level changes during the two-year period between elections. In our model, the non-linear gain-loss function quantifies the conversion between collaboration and polarization among legislators. The model implies that the polarization level always converges to an equilibrium point. We derive an approximate analytic expression for the equilibrium points, which are defined by the polarization utility and the system's sensitivity to the current polarization. Our model implies that the growing polarization utility causes the observed increased polarization in the recent few decades.

CHAPTER 5
CONCLUSIONS

5.1 Conclusions

This thesis centers around the theory and practice of community structures detection at scale and its applications to understanding the dynamics of social networks.

In this thesis, we propose several key enhancement to the modularity-based community detection algorithms. Despite being one of the most widely used state-of-the-art community detection approaches, modularity maximization suffers from the resolution limit problem which arises due to the implicit dependence of the modularity definition on a constant (explicitly defined in the generalized modularity as a resolution parameter). In this thesis, we establish the asymptotic theoretical upper and lower bounds on the resolution parameter of generalized modularity, which is the first result connecting the resolution limit of modularity with the random graph models. We also propose a progressive agglomerative heuristic algorithm that systematically increases the resolution parameter to partition a graph recursively. The statistical hypothesis testing checks if the partition found by each branch of the recursion is significant. If it is, this branch continues splitting the current graph; otherwise, the recursion branch terminates, accepting the null hypothesis that the current subgraph is already a community. In this thesis, we also show that the appropriately assigned edge weights can improve the quality of the detected communities. We propose an edge weighting scheme which improve the state-of-the-art approaches significantly and a generative random graph model which puts a constraint on nodes' internal degree ratio to stabilize the inference of block model, by preventing the inference algorithms, like Markov chain Monte Carlo, from getting trapped in the local optima of the log-likelihood.

In the context of information propagation, the community structures play an essential role in facilitating the local spread of information because the community members are more likely to accept inputs from each other than from the outsiders. Based on the statistical survival analysis, our community-affiliated information cascade model outperforms the feature-based approaches by almost 20% on the Global Database of Events, Language, and Tone (GDELT) dataset as measured by the F1-scores of the predictions. We parallelize the corresponding graph representation learning algorithm for both shared memory and dis-

tributed memory machines. The parallelized stochastic gradient descent algorithm is shown to scale well to 10K+ cores in the IBM Blue Gene/Q supercomputer at RPI (ranked among the fastest academic-owned IBM Blue Gene supercomputers).

Finally, we study the patterns of the polarization evolution by analyzing millions of roll-call votes in the legislative branches of the United States. We proposed an agent-based social dynamical model explaining the formation of polarization observed in real legislative voting data and identifying the tipping points of this dynamical system. The stable state getting close to either full polarization or full consensus provides early warning signals that the system is close to the bifurcation. Our dynamical model successfully explains the directions of polarization change in 28 out of the past 30 U.S. Congresses. The hidden variable in the dynamical model, called the polarization utility, is shown to correlate with critical historical events such as the civil rights movement and Super PACs.

REFERENCES

[1] W. O. Kermack and A. G. McKendrick, "A contribution to the mathematical theory of epidemics," in *Proc. Royal Soci. London A: Mathematical, Physical and Engineering Sci.*, 1927, vol. 115, pp. 700–721.

[2] L. J. Allen, "Some discrete-time SI, SIR, and SIS epidemic models," *Math. Biosci.*, vol. 124, no. 1, pp. 83–105, 1994.

[3] J. Li and Z. Ma, "Qualitative analyses of sis epidemic model with vaccination and varying total population size," *Math. and Comput. Modell.*, vol. 35, no. 11, pp. 1235–1243, 2002.

[4] B. A. Huberman, D. M. Romero, and F. Wu, "Social networks that matter: Twitter under the microscope," 2008, *arXiv:0812.1045v1.*

[5] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, no. 6, p. 066111, 2004.

[6] M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE Trans. Comput. Social Sys.*, vol. 1, no. 1, pp. 46–65, 2014.

[7] X. Lu, K. Kuzmin, M. Chen, and B. K. Szymanski, "Adaptive modularity maximization via edge weighting scheme," *Inf. Sci.*, vol. 424, pp. 55–68, 2018.

[8] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Phys. Rev. E*, vol. 74, no. 1, p. 016110, 2006.

[9] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proc. Nat. Acad. Sci.*, vol. 104, no. 1, pp. 36–41, 2007.

[10] X. Lu and B. K. Szymanski, "Asymptotic resolution bounds of generalized modularity and statistically significant community detection," 2019, *arXiv:1902.04243.*

[11] X. L u and B. K. Szymanski, "Regularized stochastic block model for robust community detection in complex networks," submitted for publication.

[12] A. Saxena, S. Iyengar, and Y. Gupta, "Understanding spreading patterns on social networks based on network topology," in *2015 IEEE/ACM Int. Conf. Adv. Social Netw. Anal. and Mining (ASONAM)*, 2015, pp. 1616–1617.

[13] X. Lu and B. K. Szymanski, "Predicting viral news events in online media," *Parallel and Distrib. Process. Symp. Workshops (IPDPSW)*, pp. 1447–1456, 2017.

[14] X. Lu and B. Szymanski, "Scalable prediction of global online media news virality," *IEEE Trans. Computat. Social Sys.*, no. 99, pp. 1–13, 2018.

[15] X. Lu, E. Bulut, and B. Szymanski, "Towards limited scale-free topology with dynamic peer participation," *Comput. Netw.*, vol. 106, pp. 109–121, 2016.

[16] F. Jin *et al.*, "Misinformation propagation in the age of twitter." *IEEE Comp.*, vol. 47, no. 12, pp. 90–94, 2014.

[17] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *J. Economic Perspectives*, vol. 31, no. 2, pp. 211–36, 2017.

[18] R. K. Garrett, "Echo chambers online?: Politically motivated selective exposure among internet news users," *J. Computer-Mediated Commun.*, vol. 14, no. 2, pp. 265–285, 2009.

[19] N. J. Stroud, "Polarization and partisan selective exposure," *J. Comm.*, vol. 60, no. 3, pp. 556–576, 2010.

[20] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Ann. Rev. Sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[21] A. K. Dixit and J. W. Weibull, "Political polarization," *Proc. Nat. Acad. Sci.*, vol. 104, no. 18, pp. 7351–7356, 2007.

[22] M. Deutsch and H. B. Gerard, "A study of normative and informational social influences upon individual judgment." *The J. Abnormal and Social Psychol.*, vol. 51, no. 3, p. 629, 1955.

[23] X. Lu, J. Gao, and B. K. Szymanski, "The evolution of polarization in the legislative branch of government," submitted for publication.

[24] M. E. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci.*, vol. 103, no. 23, pp. 8577–8582, 2006.

[25] A. Lancichinetti and S. Fortunato, "Limits of modularity maximization in community detection," *Phys. Rev. E*, vol. 84, no. 6, p. 066122, 2011.

[26] M. E. Newman, "Equivalence between modularity optimization and maximum likelihood methods for community detection," *Phys. Rev. E*, vol. 94, no. 5, p. 052315, 2016.

[27] B. Karrer and M. E. Newman, "Stochastic blockmodels and community structure in networks," *Phys. Rev. E*, vol. 83, no. 1, p. 016107, 2011.

[28] P. J. Bickel and A. Chen, "A nonparametric view of network models and newman–girvan and other modularities," *Proc. Nat. Acad. Sci.*, vol. 106, no. 50, pp. 21 068–21 073, 2009.

[29] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Ann. Math. Stat.*, vol. 22, no. 1, pp. 79–86, 1951.

[30] B. H. Good, Y.-A. de Montjoye, and A. Clauset, "Performance of modularity maximization in practical contexts," *Phys. Rev. E*, vol. 81, no. 4, p. 046106, 2010.

[31] S. Wagner and D. Wagner, *Comparing Clusterings: an Overview*. Karlsruhe, Germany: Universitat Karlsruhe, Fakultat fur Informatik, 2007.

[32] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mac. Learn. Res.*, vol. 11, no. Oct, pp. 2837–2854, 2010.

[33] L. Hubert and P. Arabie, "Comparing partitions," *J. Classification*, vol. 2, no. 1, pp. 193–218, 1985.

[34] M. Molloy and B. Reed, "A critical point for random graphs with a given degree sequence," *Random Structures & Algorithms*, vol. 6, no. 2-3, pp. 161–180, 1995.

[35] D. J. Fenn, M. A. Porter, M. McDonald, S. Williams, N. F. Johnson, and N. S. Jones, "Dynamic communities in multichannel data: An application to the foreign exchange market during the 2007–2008 credit crisis," *Chaos: Interdisciplinary J. Nonlinear Sci.*, vol. 19, no. 3, p. 033119, 2009.

[36] V. A. Traag, G. Krings, and P. Van Dooren, "Significant scales in community structure," *Sci. Rep.*, vol. 3, p. 2930, 2013.

[37] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela, "Community structure in time-dependent, multiscale, and multiplex networks," *Sci.*, vol. 328, no. 5980, pp. 876–878, 2010.

[38] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropological Res.*, vol. 33, no. 4, pp. 452–473, 1977.

[39] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behav. Ecol. and Sociobiol.*, vol. 54, no. 4, pp. 396–405, 2003.

[40] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, p. 026113, 2004.

[41] P. ErdHos and A. Renyi, "On the evolution of random graphs," *Publ. Math. Inst. Hung. Acad. Sci.*, vol. 5, no. 1, pp. 17–60, 1960.

[42] J. K. Ghosh and P. K. Sen, "On the asymptotic performance of the log-likelihood ratio statistic for the mixture model and related results," *In Proc. Berkeley Conf.*, pp. 789–806, 1984.

[43] S. S. Wilks, "The large-sample distribution of the likelihood ratio for testing composite hypotheses," *The Ann. Math. Statist.*, vol. 9, no. 1, pp. 60–62, 1938.

[44] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, no. 4, p. 046110, 2008.

[45] T. S. Evans, "Clique graphs and overlapping communities," *J. Statist. Mech.: Theory and Exp.*, vol. 2010, no. 12, p. P12037, 2010.

[46] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, no. 3, pp. 75–174, 2010.

[47] S. P. Borgatti and M. G. Everett, "Models of core/periphery structures," *Social Netw.*, vol. 21, no. 4, pp. 375–395, 2000.

[48] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborova, "Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications," *Phys. Rev. E*, vol. 84, no. 6, p. 066106, 2011.

[49] N. M. Nasrabadi, "Pattern recognition and machine learning," *J. Electronic Imaging*, vol. 16, no. 4, p. 049901, 2007.

[50] T. P. Peixoto, "Efficient monte carlo and greedy heuristic for the inference of stochastic block models," *Phys. Rev. E*, vol. 89, no. 1, p. 012804, 2014.

[51] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Statist. Mech.: Theory and Exp.*, vol. 2008, no. 10, p. P10008, 2008.

[52] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E*, vol. 69, no. 6, p. 066133, 2004.

[53] M. Newman, "Spectral methods for community detection and graph partitioning," *Phys. Rev. E*, vol. 88, no. 4, p. 042822, 2013.

[54] M. Sales-Pardo, R. Guimera, A. A. Moreira, and L. A. N. Amaral, "Extracting the hierarchical organization of complex systems," *Proc. Nat. Acad. Sci.*, vol. 104, no. 39, pp. 15 224–15 229, 2007.

[55] S. White and P. Smyth, "A spectral clustering approach to finding communities in graphs," in *Proc. 2005 SIAM Int. Conf. Data Mining*, 2005, pp. 274–285.

[56] M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E*, vol. 74, no. 3, p. 036104, 2006.

[57] T. Richardson, P. J. Mucha, and M. A. Porter, "Spectral tripartitioning of networks," *Phys. Rev. E*, vol. 80, no. 3, p. 036111, 2009.

[58] J. Ruan and W. Zhang, "Identifying network communities with a high resolution," *Phys. Rev. E*, vol. 77, no. 1, p. 016104, 2008.

[59] S. Fortunato and D. Hric, "Community detection in networks: A user guide," *Phys. Rep.*, vol. 659, pp. 1–44, 2016.

[60] M. Chen, T. Nguyen, and B. K. Szymanski, "A new metric for quality of network community structure," *ASE Human J.*, vol. 2, no. 4, pp. 226–240, 2012.

[61] J. W. Berry, B. Hendrickson, R. A. LaViolette, and C. A. Phillips, "Tolerating the community detection resolution limit with edge weighting," *Phys. Rev. E*, vol. 83, no. 5, p. 056119, 2011.

[62] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social Netw.*, vol. 5, no. 2, pp. 109–137, 1983.

[63] J. Nocedal and S. Wright, *Numerical Optimization*. Berlin, Germany: Springer Science & Business Media, 2006.

[64] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E*, vol. 76, no. 3, p. 036106, 2007.

[65] P. Pons and M. Latapy, "Computing communities in large networks using random walks," in *Int. Symp. Comput. and Inf. Sci.*, 2005, pp. 284–293.

[66] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. and Inf. Sys.*, vol. 42, no. 1, pp. 181–213, 2015.

[67] L. Weng, F. Menczer, and Y.-Y. Ahn, "Predicting successful memes using network and community structure," 2014, *arXiv:1403.6199*.

[68] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, "Seismic: A self-exciting point process model for predicting tweet popularity," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery and Data Mining*, 2015, pp. 1513–1522.

[69] A. Guille and H. Hacid, "A predictive model for the temporal dynamics of information diffusion in online social networks," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 1145–1152.

[70] A. Nematzadeh, E. Ferrara, A. Flammini, and Y.-Y. Ahn, "Optimal network modularity for information diffusion," *Phys. Rev. Letters*, vol. 113, no. 8, p. 088701, 2014.

[71] P. Bogdanov, M. Busch, J. Moehlis, A. K. Singh, and B. K. Szymanski, "Modeling individual topic-specific behavior and influence backbone networks in social media," *Social Netw. Anal. and Mining*, vol. 4, no. 1, pp. 1–16, 2014.

[72] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Adv. Neural Inf. Process. Sys.*, 2001, pp. 556–562.

[73] I. Psorakis, S. Roberts, M. Ebden, and B. Sheldon, "Overlapping community detection using bayesian non-negative matrix factorization," *Phys. Rev. E*, vol. 83, no. 6, p. 066114, 2011.

[74] J. Yang and J. Leskovec, "Overlapping community detection at scale: a nonnegative matrix factorization approach," in *Proc. 6th ACM Int. Conf. Web Search and Data Mining*, 2013, pp. 587–596.

[75] M. Gomez Rodriguez, J. Leskovec, and B. Scholkopf, "Structure and dynamics of information pathways in online media," in *Proc. 6th ACM Int. Conf. Web Search and Data Mining*, 2013, pp. 23–32.

[76] R. Albert, H. Jeong, and A.-L. Barabasi, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.

[77] G. Korniss, "Synchronization in weighted uncorrelated complex networks in a noisy environment: Optimization and connections with transport efficiency," *Phys. Rev. E*, vol. 75, no. 5, p. 051121, 2007.

[78] Z. Toroczkai and K. E. Bassler, "Network dynamics: Jamming is limited in scale-free systems," *Nature*, vol. 428, no. 6984, pp. 716–716, 2004.

[79] A.-L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Sci.*, vol. 286, no. 5439, pp. 509–512, 1999.

[80] H. Guclu and M. Yuksel, "Limited scale-free overlay topologies for unstructured peer-to-peer networks," *IEEE Trans. Parallel and Distrib. Sys.*, vol. 20, no. 5, pp. 667–679, 2009.

[81] G. Bent, P. Dantressangle, D. Vyvyan, A. Mowshowitz, and V. Mitsou, "A dynamic distributed federated database," in *Proc. 2nd Ann. Conf. Int. Technol. Alliance*, 2008.

[82] E. Bulut and B. K. Szymanski, "Constructing limited scale-free topologies over peer-to-peer networks," *IEEE Trans. Parallel and Distrib. Sys.*, vol. 25, no. 4, pp. 919–928, 2014.

[83] K. Leetaru and P. A. Schrodt, "Gdelt: Global data on events, location, and tone, 1979–2012," in *ISA Annu. Convention*, 2013, vol. 2, pp. 1–49.

[84] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[85] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *Proc. ninth ACM SIGKDD Int. Conf. Knowl. Discovery and Data Mining*, 2003, pp. 137–146.

[86] A.-L. Barabasi, "The origin of bursts and heavy tails in human dynamics," *Nature*, vol. 435, no. 7039, pp. 207–211, 2005.

[87] S. Hjarvard, "News media and the globalization of the public sphere," *News in a Globalized Soc.*, pp. 17–39, 2001.

[88] J. Xie and B. K. Szymanski, "Community detection using a neighborhood strength driven label propagation algorithm," in *Network Sci. Workshop (NSW), 2011 IEEE*, 2011, pp. 188–195.

[89] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," 2012, *arXiv:1206.6426*.

[90] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Adv. Neural Inf. Process. Sys.*, 2013, pp. 3111–3119.

[91] S. Ji, N. Satish, S. Li, and P. Dubey, "Parallelizing word2vec in shared and distributed memory," 2016, *arXiv:1604.04661*.

[92] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Adv. Neural Inf. Process. Sys.*, 2011, pp. 693–701.

[93] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-passing Interface*. Cambridge, MA, USA: MIT press, 1999, vol. 1.

[94] G. Malewicz *et al.*, "Pregel: a system for large-scale graph processing," in *Proc. 2010 ACM SIGMOD Int. Conf. Manage. Data*, 2010, pp. 135–146.

[95] C. Avery, "Giraph: Large-scale graph process. infrastructure on hadoop," *Proc. Hadoop Summit. Santa Clara*, vol. 11, no. 3, pp. 5–9, 2011.

[96] R. Haring *et al.*, "The ibm blue gene/q computer chip," *IEEE Micro*, vol. 32, no. 2, pp. 48–60, 2012.

[97] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. and Prob.*, 1967, vol. 1, no. 14, pp. 281–297.

[98] A. Clauset, C. R. Shalizi, and M. E. Newman, "Power-law distributions in empirical data," *SIAM Rev.*, vol. 51, no. 4, pp. 661–703, 2009.

[99] T. F. Cox and M. A. Cox, *Multidimensional Scaling*. London, UK: Chapman and Hall/CRC, 2000.

[100] A. Liaw, M. Wiener, Y. X, and Z. W, "Classification and regression by random forest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[101] H. Jeong, S. P. Mason, A.-L. Barabasi, and Z. N. Oltvai, "Lethality and centrality in protein networks," *Nature*, vol. 411, no. 6833, pp. 41–42, 2001.

[102] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *ACM SIGCOMM Comput. Commun. Rev.*, 1999, vol. 29, pp. 251–262.

[103] A.-L. Barabasi, R. Albert, and H. Jeong, "Diameter of the world-wide web," *Nature*, vol. 401, no. 9, pp. 130–131, 1999.

[104] A.-L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek, "Evolution of the social network of scientific collaborations," *Phys. A: Statist. Mech. and its Appl.*, vol. 311, no. 3, pp. 590–614, 2002.

[105] B. K. Szymanski and G. G. Chen, "Computing with time: from neural networks to sensor networks," *Comput. J.*, vol. 51, no. 4, pp. 511–522, 2008.

[106] B. Szymanski, "On growing perfect power-law graphs," Dep. Comput. Sci., Rensselaer Polytechnic Inst., Troy, NY, USA, Tech. Rep. 11-02, 2011.

[107] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, "Search in power-law networks," *Phys. Rev. E*, vol. 64, no. 4, p. 046135, 2001.

[108] L. J. Diamond, "Three paradoxes of democracy," *J. Democracy*, vol. 1, no. 3, pp. 48–60, 1990.

[109] J. C. Turner, *Social Influence*. Belmont, CA, USA: Thomson Brooks/Cole Publishing Co, 1991.

[110] S. Feldman, "Enforcing social conformity: A theory of authoritarianism," *Political Psychol.*, vol. 24, no. 1, pp. 41–74, 2003.

[111] V. Klucharev, K. Hytonen, M. Rijpkema, A. Smidts, and G. Fernandez, "Reinforcement learning signal predicts social conformity," *Neuron*, vol. 61, no. 1, pp. 140–151, 2009.

[112] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Commun. ACM*, vol. 59, no. 7, pp. 96–104, 2016.

[113] J. K. Madsen, R. M. Bailey, and T. D. Pilditch, "Large networks of rational agents form persistent echo chambers," *Sci. Rep.*, vol. 8, no. 1, p. 12391, 2018.

[114] K. Garimella, G. D. F. Morales, A. Gionis, and M. Mathioudakis, "Quantifying controversy on social media," *ACM Trans. Social Comput.*, vol. 1, no. 1, p. 3, 2018.

[115] J. B. Lewis, K. Poole, H. Rosenthal, A. Boche, A. Rudkin, and L. Sonnet, "Voteview: Congressional roll-call votes database," 2017.

[116] R. Jurdak, K. Zhao, J. Liu, M. AbouJaoude, M. Cameron, and D. Newth, "Understanding human mobility from twitter," *PloS ONE*, vol. 10, no. 7, p. e0131469, 2015, doi: 10.1371/journal.pone.0131469.

[117] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Sci.*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[118] S. Ansolabehere, A. Gerber, and J. M. Snyder Jr, *Corruption and the Growth of Campaign Spending: a Users Guide to Campaign Finance Reform*. Lanham, MD, USA: Rowman and Littlefield, 2001.

[119] D. Baldassarri and P. Bearman, "Dynamics of political polarization," *American Sociol. Rev.*, vol. 72, no. 5, pp. 784–811, 2007.

[120] S. A. Rice, "Quantitative methods in politics." *J. American Statist. Assoc.*, 1928.

[121] J. Moody and P. J. Mucha, "Portrait of political party polarization," *Net. Sci.*, vol. 1, no. 1, pp. 119–121, 2013.

[122] Y. Gu, Y. Sun, and J. Gao, "The co-evolution model for social network evolving and opinion migration," in *Proc. 23rd ACM SIGKDD Int. Conf. on Knowl. Discovery and Data Mining*, 2017, pp. 175–184.

[123] H. Läuter, "Silverman, bw: Density estimation for statistics and data analysis," *Biometrical J.*, vol. 30, no. 7, pp. 876–877, 1988.

[124] D. M. Abrams, H. A. Yaple, and R. J. Wiener, "Dynamics of social group competition: modeling the decline of religious affiliation," *Phys. Rev. Letters*, vol. 107, no. 8, p. 088701, 2011.

[125] D. M. Abrams and S. H. Strogatz, "Linguistics: Modelling the dynamics of language death," *Nature*, vol. 424, no. 6951, p. 900, 2003.

[126] L. Frachebourg and P. Krapivsky, "Exact results for kinetics of catalytic reactions," *Phys. Rev. E*, vol. 53, no. 4, p. R3009, 1996.

[127] L. Perko, *Differential Equations and Dynamical Systems*. Berlin, Germany: Springer Science & Business Media, 2013, vol. 7.

[128] C. A. Bail *et al.*, "Exposure to opposing views on social media can increase political polarization," *Proc. Nat. Acad. Sci.*, vol. 115, no. 37, pp. 9216–9221, 2018.

[129] D. Baldassarri and A. Gelman, "Partisans without constraint: Political polarization and trends in american public opinion," *American J. Sociology*, vol. 114, no. 2, pp. 408–446, 2008.