Chapter 4

COMMUNITY DISCOVERY IN SOCIAL NETWORKS: APPLICATIONS, METHODS AND EMERGING TRENDS

S. Parthasarathy
The Ohio State University
2015 Neil Ave, DL395, Columbus, OH
srini@cse.ohio-state.edu

Y. Ruan
The Ohio State University
2015 Neil Ave, DL395, Columbus, OH
ruan@cse.ohio-state.edu

V. Satuluri
The Ohio State University
2015 Neil Ave, DL395, Columbus, OH
satuluri@cse.ohio-state.edu

Abstract Data sets originating from many different real world domains can be represented in the form of interaction networks in a very natural, concise and meaningful fashion. This is particularly true in the social context, especially given recent advances in Internet technologies and Web 2.0 applications leading to a diverse range of evolving social networks. Analysis of such networks can result in the discovery of important patterns and potentially shed light on important properties governing the growth of such networks.

It has been shown that most of these networks exhibit strong modular nature or community structure. An important research agenda thus is to identify communities of interest and study their behavior over time. Given the importance of this problem there has been significant activity within this field particularly over the last few years. In this article we survey the landscape and attempt to characterize the principle methods for community discovery (and related variants)

C. C. Aggarwal (ed.), *Social Network Data Analytics,*
DOI 10.1007/978-1-4419-8462-3_4,

and identify current and emerging trends as well as crosscutting research issues within this dynamic field.

Keywords: Graph Mining, Community Discovery, Social Networks

1. Introduction

Many real world problems can be effectively modeled as complex relationship networks where nodes represent entities of interest and edges mimic the interactions or relationships among them. Fueled by technological advances and inspired by empirical analysis, the number of such problems and the diversity of domains from which they arise – biological [47, 86, 98, 114, 118], clinical [9, 30], ecological [14, 33, 68, 69, 97], engineering [1, 75, 122], linguistic [46], scientific [29, 42, 71, 99], social [72, 91, 113, 121, 124], technological [4, 34, 37, 87, 92], to name a few – is growing steadily.

It has recently been observed that while such networks arise in a host of diverse arenas they often share important common concepts or themes. The study of such complex relationship networks, recently referred to as *network science*, can provide insight into their structures, properties and emergent behaviors [11, 12, 41, 77]. Of particular interest here are rigorous methods for uncovering and understanding important network or sub-network (community) structures and motifs at multiple topological and temporal scales as noted in a recent government report [17]. Extracting such community structure and leveraging them for predicting the emergent, critical, and causal nature of such networks in a dynamic setting is of growing importance.

Extracting such structure is indeed a grand challenge. First, the topological properties of such networks coupled with an uncertain setting [5, 102], often limit the applicability of existing off-the-shelf techniques [9]. Second, the requirements imposed by directed and dynamic [1] networks require research into appropriate solutions. Finally, underpinning all of these challenges is the issue of scalability. Many of the problems we consider require us to deal with problems of immense size and scale where graphs may involve millions of nodes and billions of edges[48].

In this chapter we limit our discussion primarily to the problem of community detection within social networks (albeit a lot of what we will discuss may apply naturally to other domains as well). We begin by discussing why

[1] By *dynamic*, here we refer to any network that changes. This includes not only time-varying networks, but also networks that change due to external factors (e.g. networks that change due to trust issues and source credibility issues, such as intelligence networks).

community discovery in such networks is useful in Section 2. In other words what are the actionable patterns[82] or tools one can derive from such an analysis on social networks. Sample applications abound ranging from the study of intelligence reports to the social behavior of Zebras and Dolphins, from the collaborative nature of physical and computer scientists, to the often cited Karate Club social network, from well established communities in Facebook to the role of communities in Twitter networks for emergency management. We discuss these issues in Section 2.

In Section 3 we discuss the core methods for community discovery proposed in the literature to date. We discuss hierarchical algorithms (agglomerative or divisive) that are popular within the Physics community [78]. The advantages of such algorithms lie in their intuitive simplicity but as noted elsewhere [24] they often do not scale well to large networks. We take a close look at the related literature in graph partitioning, starting with the early work by Kernighan and Lin, as well as more recent multi-level graph partitioning algorithms such as Metis [51], Graclus and MLR-MCL[93]. These are highly scalable and have been used in studies of some of the biggest graph datasets [61, 93]. Spectral methods that target weighted cuts [96] form an important class of algorithms that can be used for community discovery, and are shown to be qualitatively very effective in the social network context. Recent advances in this domain have targeted large scale networks (e.g. local spectral clustering) and these will be discussed as well. To this mix of graph clustering and community discovery algorithms one can also include Markov Clustering (MCL), a graph clustering algorithm based on (stochastic) flow simulation [115]. MCL has drawn limited attention from the broader network science, web science, and data mining communities primarily because it does not scale very well even to moderate sized graphs [24], and other limitations. However, recent advances have suggested effective ways to redress these limitations while retaining its advantages[93, 95]. In addition to the above recent research has suggested the use of hybrid algorithms (e.g. Metis+MQI) and the notion of different kinds of community structures (e.g. whiskers and viewpoint neighborhoods), and we will discuss these in Section 3 as well.

In Section 4 we will primarily discuss relatively new domains within social network analysis where community discovery can play an important role as we move forward. Particular attention here will be paid to work on community discovery in heterogeneous social networks (e.g. Flickr where links may correspond to common tags, similar images, or similar user profiles), community discovery in dynamic social networks (community evolution, dispersion, merging[9]), community discovery in directed social networks (e.g. Twitter), and community discovery that combines content and network information in a natural manner (e.g. topic driven community discovery, social media analytics

etc.). In Section 5, we conclude with a discussion of cross cutting research issues that relate to the current state-of-art in this area.

2. Communities in Context

In this section we will discuss the role(s) of community discovery in social network analysis. Specifically we will discuss end applications and contextual benefits from both a scientific as well as actionable perspective.

Social community analysis has been the focus of many studies over the past eighty years[121]. One of the earliest studies in this context include work by Rice on the analysis of communities of individuals based on their political biases and voting patterns[89]. A much more recent study along similar lines but focusing on the network structure of political blogs was discussed by Adamic and Glance[2]. Homans was the first to show that social groups could be revealed by suitably rearranging the rows and the columns of matrices describing social ties, until they take an approximate block-diagonal form[44]. In fact this idea still serves as the basic tool for visualizing social community structure and more generally clustering structure[110]. Weiss and Jacobson examined work groups within a government agency[123]. A central theme of their work was the identification of bridging nodes and using such nodes to separate out community structure. In fact this work can be thought of as an early version of the notion of betweenness centrality popularized by Newman[76]. The karate club study by anthropologist Zachary, is a well-known graph regularly commonly used as a benchmark to test community detection algorithms[126]. It consists of 34 vertices, the members of a karate club in the United States, who were observed during a period of three years and it includes a well known community fission instance and thus the subject of many studies. Another study by Bech and Atalay analyzed the social network of loans among financial institutions to understand how interactions among multiple communities affect the health of the system as a whole[15]. A large majority of these studies focused on simply understanding the underpinning social structure and its evolution (for instance in the karate club data the underlying cause of the fission in the community structure resulting from a difference of opinion between two members of the club).

In addition to the study of human social networks zoologists and biologists have also begun to study the social behavior of other animals and sea creatures. Lusseau in a land mark study examined the behavior of 62 bottlenose dolphins off the coast of New Zealand[65]. This study looked at the social behavior of 62 dolphins and edges were set between animals that were seen together more often than expected by random chance. Lusseau notes the cohesive and cliquish structure of the resulting graph suggesting that the social behavior of such marine mammals is often quite marked. More recently an in-

terdisciplinary team comprising zoologists and computer scientists have studied the social behavior of zebras[109]. Insights ranging from the identification of influential herd leaders within communities, and the evolutionary behavior of the resulting social network of zebras have led to a significantly increased understanding of how these animals communicate and socialize to survive.

Since the advent of the Internet and more recently the World Wide Web a number of applications of community discovery have arisen. In the World Wide Web context a common application of community discovery is in the context of proxy caches[103]. A grouping of web clients who have similar interests and are geographically near each other may enable them to be served by a dedicated proxy server. Another example from the same domain is to identify communities within the hyperlinked structure of the web. Such communities may help in the detection of link farms[21, 40]. Similarly in the E-commerce domain the grouping together of customers with similar buying profiles enables more personalized recommendation engines[88]. In a completely different arena, community discovery in mobile ad-hoc networks can enable efficient message routing and posting[104]. In this context it is important to distinguish core members of the community from members on the border (analogous to edge routers).

Recently there has been a tremendous thrust in the use of community discovery techniques for analyzing online social media data[73]. The user-generated content explosion on Web 2.0 applications such as Twitter, Facebook, review blogs, micro-blogs and various multimedia sharing sites such as Flickr, presents many opportunities for both facilitators and users. For facilitators, this user-generated content is a rich source of implicit consumer feedback. For users the ability to sense and respond interactively and to be able to leverage the wisdom of the crowds (or communities) can be extremely fruitful and useful. It is becoming increasingly clear that a unified approach to analysis combining content information with network analysis is necessary to make headway into this arena.

The above examples show that community discovery in social or socio-technical networks is at the heart of various research agendas. We are now in a position to discuss some of the implications of this technique. At the most fundamental level, community discovery (either in a static or evolutionary context) can facilitate and aid in our understanding of a social system. Much like the role of clustering and community discovery can be thought of as clustering on graphs, community discovery allows us to summarize the interactions within a network concisely, enabling a richer understanding of the underlying social phenomenon. Beyond this basic understanding of the network and how it evolves[9], community discovery can also lend itself to actionable pattern discovery. Identification of influential nodes, or sub-communities within a broader community can be used for viral marketing[32, 53, 59], churn predic-

tion within telecommunication networks[90] and ratings predictions[56]. We will conclude this section with an example of how community discovery can be useful for organizational entities.

Von Hayek, one of the leading economists of the twentieth century, when discussing a competitive market mechanism, articulated the important fact that the minds of millions of people is not available to any central body or any group of decision-makers who have to determine prices, employment, production, and investment policies[116]. He further argued that an increase in decentralization was an essential component to rational decision-making by organizations in a complex society. The ideas he expounded have broad utility beyond the context he was considering. Emergency management is an analogous example of a large and complex socio-technical system, where many people distributed in space and time may potentially harness the power of modern social information technologies to coordinate activities of many in order to accomplish a complex task. In this context recent work by Palen and Liu [80] makes a strong case for the value of understanding the dynamic of social networking and specifically community structure, in relation to managing and mitigating the impact of disasters.

3. Core Methods

Informally, a community in a network is a group of nodes with greater ties internally than to the rest of the network. This intuitive definition has been formalized in a number of competing ways, usually by way of a quality function, which quantifies the goodness of a given division of the network into communities. Some of these quality metrics, such as Normalized Cuts [96] and Modularity [78] are more popular than others, but none has gained universal acceptance since no single metric is applicable in all situations. Several such metrics are discussed in Section 3.1.

Algorithms for community discovery vary on a number of important dimensions, including their approach to the problem as well as their performance characteristics. An important dimension on which algorithms vary in their approaches is whether or not they explicitly optimize a specific quality metric. Spectral methods, the Kernighan-Lin algorithm and flow-based postprocessing are all examples of algorithms which explicitly try to optimize a specific quality metric, while other algorithms, such as Markov Clustering (MCL) and clustering via shingling do not do so. Another dimension on which algorithms vary is in how (or even whether) they let the user control the granularity of the division of the network into communities. Some algorithms (such as spectral methods) are mainly meant for bi-partitioning the network, but this can be used to recursively subdivide the network into as many communities as desired. Other algorithms such as agglomerative clustering or MCL allow the

user to indirectly control the granularity of the output communities through certain parameters. Still other algorithms, such as certain algorithms optimizing the Modularity function, do not allow (or require) the user to control the output number of communities at all. Another important characteristic differentiating community discovery algorithms is the importance they attach to a balanced division of the network - while metrics such as the KL objective function explicitly encourage balanced division, other metrics capture balance only implicitly or not at all. Coming to performance characteristics, algorithms also vary in their scalability to big networks, with multi-level clustering algorithms such as Metis, MLR-MCL and Graclus and local clustering algorithms scaling better than many other approaches.

3.1 Quality Functions

A variety of quality functions or measures have been proposed in the literature to capture the goodness of a division of a graph into clusters. In what follows, A denotes the adjacency matrix of the network or graph, with $A(i, j)$ representing the edge weight or affinity between nodes i and j, and V denotes the vertex or node set of the graph or network.

The normalized cut of a group of vertices $S \subset V$ is defined as[96, 67]

$$Ncut(S) = \frac{\sum_{i \in S, j \in \bar{S}} A(i,j)}{\sum_{i \in S} degree(i)} + \frac{\sum_{i \in S, j \in \bar{S}} A(i,j)}{\sum_{j \in \bar{S}} degree(j)} \tag{4.1}$$

In words, the normalized cut of a group of nodes S is the sum of weights of the edges that connect S to the rest of the graph, normalized by the total edge weight of S and that of the rest of the graph $\bar{S}$. Intuitively, groups with low normalized cut make for good communities, as they are well connected amongst themselves but are sparsely connected to the rest of the graph.

The *conductance* of a group of vertices $S \subset V$ is closely related and is defined as [50]

$$Conductance(S) = \frac{\sum_{i \in S, j \in \bar{S}} A(i,j)}{\min(\sum_{i \in S} degree(i), \sum_{i \in \bar{S}} degree(i))} \tag{4.2}$$

The normalized cut (or conductance) of a division of the graph into k clusters $V_1, \ldots, V_k$ is the sum of the normalized cuts (or conductances) of each of the clusters $V_i \{i = 1, \ldots, k\}$ [31].

The *Kernighan-Lin (KL) objective* looks to minimize the edge cut (or the sum of the inter-cluster edge weights) under the constraint that all clusters be of the same size (making the simplifying assumption that the size of the

network is a multiple of the number of clusters):

$$KLObj(V_1, \ldots, V_k) = \sum_{i \neq j} A(V_i, V_j) \text{subject to} |V_1| = |V_2| = \ldots = |V_k| \tag{4.3}$$

Here $A(V_i, V_j)$ denotes the sum of edge affinities between vertices in V_i and V_j, i.e. $A(V_i, V_j) = \sum_{u \in V_i, v \in V_j} A(u, v)$

Modularity [78] has recently become quite popular as a way to measure the goodness of a clustering of a graph. One of the advantages of modularity is that it is independent of the number of clusters that the graph is divided into. The intuition behind the definition of modularity is that the farther the subgraph corresponding to each community is from a random subgraph (i.e. the null model), the better or more significant the discovered community structure is. The modularity Q for a division of the graph into k clusters $\{V_1, \ldots, V_k\}$ is given by:

$$Q = \sum_{c=1}^{k} \left[\frac{A(V_i, V_i)}{m} - \left(\frac{degree(V_i)}{2m} \right)^2 \right] \tag{4.4}$$

In the above, the V_is are the clusters, m is the number of edges in the graph and $degree(V_i)$ is the total degree of the cluster V_i. For each cluster, we take the difference between the fraction of edges internal to that cluster and the fraction of edges that would be expected to be inside a random cluster with the same total degree.

Optimizing any of these objective functions is NP-hard [39, 96, 18].

3.2 The Kernighan-Lin(KL) algorithm

The KL algorithm [54] is one of the classic graph partitioning algorithms which optimizes the KL objective function i.e. minimize the edge cut while keeping the cluster sizes balanced (see Equation 4.3. The algorithm is iterative in nature and starts with an initial bipartition of the graph. At each iteration, the algorithm searches for a subset of vertices from each part of the graph such that swapping them will lead to a reduction in the edge cut. The identification of such subsets is via a greedy procedure. The *gain* g_v of a vertex v is the reduction in edge-cut if vertex v is moved from its current partition to the other partition. The KL algorithm repeatedly selects from the larger partition the vertex with the largest gain and moves it to the other partition; a vertex is not considered for moving again if it has already been moved in the current iteration. After a vertex has been moved, the gains for its neighboring vertices will be updated in order to reflect the new assignment of vertices to partitions. While each iteration in the original KL algorithm [54] had a complexity of $O(|E| \log |E|)$, Fiduccia and Mattheyses improved it to $O(|E|)$ per iteration using appropriate data structures. This algorithm can be extended to multi-

way partitions by improving each pair of partitions in the multi-way partition in the above described way.

3.3 Agglomerative/Divisive Algorithms

Agglomerative algorithms begin with each node in the social network in its own community, and at each step merge communities that are deemed to be sufficiently similar, continuing until either the desired number of communities is obtained or the remaining communities are found to be too dissimilar to merge any further. Divisive algorithms operate in reverse; they begin with the entire network as one community, and at each step, choose a certain community and split it into two parts. Both kinds of hierarchical clustering algorithms often output a *dendrogram* which is a binary tree , where the leaves are nodes of the network, and each internal node is a community. In the case of divisive algorithms, a parent-child relationship indicates that the community represented by the parent node was divided to obtain the communities represented by the child nodes. In the case of agglomerative algorithms, a parent-child relationship in the dendrogram indicates that the communities represented by the child nodes were agglomerated (or merged) to obtain the community represented by the parent node.

Girvan and Newman's divisive algorithm: Newman and Girvan [78] proposed a divisive algorithm for community discovery, using ideas of *edge betweenness*. Edge betweenness measures are defined in a way that edges with high betweenness scores are more likely to be the edges that connect different communities. That is, inter-community edges are designed to have higher edge betweenness scores than intra-community edges do. Hence, by identifying and discarding such edges with high betweenness scores, one can disconnect the social network into its constituent communities.

Shortest path betweenness is one example of an edge betweenness measure: the intuitive idea here is that since there will only be a few inter-community edges, shortest paths between nodes that belong to different communities will be constrained to pass through those few inter-community edges. Also enumerated are two other examples of edge betweenness. In the definition of *random-walk betweenness*, the choice of path connecting any two nodes is the result of random walk instead of geodesic as in the case of *shortest path*. The *current-flow betweenness* definition is motivated by the circuit theory. First the network is virtually transformed into a resistance network where each edge is replaced by a unit resistance and two nodes are chosen as unit current source and sink. Then the betweenness of each edge is computed as the sum of absolute values of the currents flowing on it with all possible selections of node pairs.

The general form of their algorithms is as follows:

1 Calculate betweenness score for all edges in the network using any measure.

2 Find the edge with the highest score and remove it from the network.

3 Recalculate betweenness for all remaining edges.

4 Repeat from step 2.

The above procedure is continued until a sufficiently small number of communities are obtained, and a hierarchical nesting of the communities is also obtained as a natural by-product. On the contrary to the speculation that different measures of edge betweenness may lead to diverged community structures, the experiment showed that the exact betweenness measure used is not so crucial. As long as the recalculation step is executed, the results by different measures only differ from each other slightly. The motivation for the recalculation step is as follows: if the edge betweenness scores are only calculated once and edges are then removed by the decreasing order of scores, these scores won't get updated and no longer reflect the new network structure after edge removals. Therefore, recalculation is in fact the most critical step in the algorithm to achieve satisfactory results. The main disadvantage of this approach is the high computational cost: simply computing the betweenness for all edges takes $O(|V||E|)$ time, and the entire algorithm requires $O(|V|^3)$ time.

Newman's greedy optimization of modularity: Newman [74] proposed a greedy agglomerative clustering algorithm for optimizing modularity. The basic idea of the algorithm is that at each stage, groups of vertices are successively merged to form larger communities such that the modularity of the resulting division of the network increases after each merge. At the start, each node in the network is in its own community, and at each step one chooses the two communities whose merger leads to the biggest increase in the modularity. We only need to consider those communities which share at least one edge, since merging communities which do not share any edges cannot result in an increase in modularity - hence this step takes $O(|E|)$ time. An additional data structure which maintains the fraction of shared edges between each pair of communities in the current partition is also maintained, and updating this data structure takes worst-case $O(|V|)$ time. There are a total of $|V| - 1$ iterations (i.e. mergers), hence the algorithm requires $O(|V|^2)$ time. Clauset et al. [29] later improved the complexity of this algorithm by the use of efficient data structures such as max-heaps, with the final complexity coming to $O(|E|d \log |V|)$, where d is the depth of the dendrogram describing the successive partitions found during the execution of the algorithm.

3.4 Spectral Algorithms

Spectral algorithms are among the classic methods for clustering and community discovery. Spectral methods generally refer to algorithms that assign nodes to communities based on the eigenvectors of matrices, such as the adjacency matrix of the network itself or other related matrices. The top k eigenvectors define an embedding of the nodes of the network as points in a k-dimensional space, and one can subsequently use classical data clustering techniques such as K-means clustering to derive the final assignment of nodes to clusters [117]. The main idea behind spectral clustering is that the low-dimensional representation, induced by the top eigenvectors, exposes the cluster structure in the original graph with greater clarity. From an alternative perspective, spectral clustering can be shown to solve real relaxations of different weighted graph cut problems, including the normalized cut defined above [117, 96].

The main matrix that is used in spectral clustering applications is the Laplacian matrix $\mathcal{L}$. If A is the adjacency matrix of the network, and D is the diagonal matrix with the degrees of the nodes along the diagonal, then the unnormalized Laplacian L is given as $L = D - A$. The Laplacian (or the normalized Laplacian) $\mathcal{L}$ is given by $\mathcal{L} = D^{-1/2}(D - A)D^{-1/2} = I - D^{-1/2}AD^{-1/2}$. It can be verified that both L and $\mathcal{L}$ are symmetric and positive definite, and therefore have real and positive eigenvalues [27, 117]. The Laplacian has 0 as an eigenvalue with multiplicity equal to the number of connected components in the graph. The eigenvector corresponding to the smallest non-zero eigenvalue of $\mathcal{L}$ is known as the Fiedler vector [35], and usually forms the basis for bi-partitioning the graph.

The main disadvantage of spectral algorithms lies in their computational complexity. Most modern implementations for eigenvector computation use iterative algorithms such as the Lanczos algorithm, where at each stage a series of matrix vector multiplications are performed to obtain successive approximations to the eigenvector currently being computed. The complexity for computing the top eigenvector is $O(kM(m))$, where k is the number of matrix-vector multiplications and $M(m)$ is the complexity of each such multiplication, dependent primarily on the number of non-zeros m in the matrix. k depends on the specific properties of the matrix at hand - such as the spectral gap i.e. the difference between the current eigenvalue and the next eigenvalue; the smaller this gap, the more number of matrix-vector multiplications are required for convergence. In practice, spectral clustering is hard to scale up to networks with more than tens of thousands of vertices without employing parallel algorithms.

Dhillon et al. [31] showed that the weighted cut measures such as normalized cut that are often optimized using spectral clustering can also be opti-

mized using an equivalent weighted kernel k-means algorithm. This is the core idea behind their algorithm **Graclus**, which can cluster graphs at a comparable quality to spectral clustering without paying the same computational cost, since k-means is much faster compared to eigenvector computation.

3.5 Multi-level Graph Partitioning

Multi-level methods provide a powerful framework for fast and high-quality graph partitioning, and in fact have been used for solving a variety of other problems as well [112]. The main idea here is to shrink or coarsen the input graph successively so as to obtain a small graph, partition this small graph and then successively project this partition back up to the original graph, refining the partition at each step along the way. Multi-level graph partitioning methods include multi-level spectral clustering [13], **Metis** (which optimizes the KL objective function) [51], **Graclus** (which optimizes normalized cuts and other weighted cuts) [31] and **MLR-MCL** [93] (further discussed in Section 3.6).

The main components of a multi-level graph partitioning strategy are:

1 **Coarsening**. The goal here is to produce a smaller graph that is similar to the original graph. This step may be applied repeatedly to obtain a graph that is small enough to be partitioned quickly and with high-quality. A popular coarsening strategy is to first construct a *matching* on the graph, where a matching is defined as a set of edges no two of which are incident on the same vertex. For each edge in the matching, the vertices at the ends of the edge are collapsed together and are represented by a single node in the coarsened graph. Coarsening can be performed very quickly using simple randomized strategies [51].

2 **Initial partitioning**. In this step, a partitioning of the coarsest graph is performed. Since the graph at this stage is small enough, one may use strategies like spectral partitioning which are slow but are known to give high quality partitions.

3 **Uncoarsening**. In this phase, the partition on the current graph is used to initialize a partition on the finer (bigger) graph. The finer connectivity structure of the graph revealed by the uncoarsening is used to refine the partition, usually by performing local search. This step is continued until we arrive at the original input graph.

At a finer level, Metis uses a variant of the KL algorithm in its uncoarsening phase to refine the partition obtained from previous steps. Graclus, on the other hand, uses weighted kernel k-means for refining the partition.

3.6 Markov Clustering

Stijn van Dongen's Markov Clustering algorithm (MCL) clusters graphs via manipulation of the stochastic matrix or transition probability matrix corresponding to the graph [115]. In what follows, the transition probability between two nodes is also referred to as *stochastic flow*. The MCL process consists of two operations on stochastic matrices, *Expand* and *Inflate*. *Expand(*M*)* is simply $M * M$, and *Inflate(*M, r*)* raises each entry in the matrix M to the inflation parameter r (> 1, and typically set to 2) followed by re-normalizing the columns to sum to 1. These two operators are applied in alternation iteratively until convergence, starting with the initial transition probability matrix.

The *expand* step spreads the stochastic flow out of a vertex to potentially new vertices and also enhances the stochastic flow to those vertices which are reachable by multiple paths. This has the effect of enhancing within-cluster stochastic flows as there are more paths between two nodes that are in the same cluster than between those in different clusters. The inflation step introduces a non-linearity into the process, with the purpose of strengthening intra-cluster stochastic flow and weakening inter-cluster stochastic flow. The process as a whole sets up a positive feedback loop that forces all the nodes within a tightly-linked group of nodes to stochastically flow to one "attractor" node within the group, allowing us to identify the group.

MCL has received a lot of attention in the bioinformatics field, with multiple researchers finding it to be very effective at clustering biological interaction networks ([20, 62]). However, MCL has two major shortcomings [93]. First, MCL is slow, since the Expand step, which involves matrix-matrix multiplication, is very time consuming in the first few iterations when many entries in the stochastic flow matrix have not been pruned out. Second, MCL tends to produce imbalanced clustering, usually by producing a large number of very small clusters (singleton clusters or clusters with only 2 or 3 nodes), or by producing one very big cluster, or by doing both at the same time.

Recent Variants of MCL: Recently, Regularized MCL and Multi-level Regularized MCL (MLR-MCL) [93, 95] have been proposed that fix the above two weaknesses of poor scalability and imbalanced clustering. Regularized MCL ensures that the stochastic flows of neighboring nodes are taken into account when updating the stochastic flows of each node by replacing the *Expand* step of MCL with a *Regularize* step, which is $M := M * M_G$, where M_G is the original stochastic (transition) matrix corresponding to the graph. Other regularization matrices instead of M_G are also explored in [95] with the intention of reducing the imbalance in the sizes of output clusters. Multi-level Regularized MCL (MLR-MCL) embeds Regularized MCL in a multi-level framework, with the algorithm working its way up the chain of coarsened graphs of the input

graph, and projecting intermediate results from the smaller graph onto the next bigger graph. MLR-MCL achieves state-of-the-art scalability since the initial iterations of the algorithm, which are the most expensive in the total computation, are performed on the smallest graphs, and the matrices are sparse enough at the biggest graphs to enable fast multiplication.

3.7 Other Approaches

Local Graph Clustering: A *local algorithm* is one that finds a solution containing or near a given vertex (or vertices) without looking at the whole graph. Local algorithms are interesting in the context of large graphs since their time complexity depends on the size of the solution rather than the size of the graph to a large extent. (Although if the clusters need to cover the whole graph, then it is not possible to be independent of the size of the graph.) The main intuition is that random walks simulated from inside a group of internally well-connected nodes will not mix well enough/soon enough, as the cluster boundary acts a bottleneck that prevents the probability from seeping out of the cluster easily. Low-probability vertices are removed at each step to keep the complexity of the algorithm tractable.

Spielman and Teng [101, 100] described the first such local clustering algorithm using random walks. Let $p_{t,v}$ be the probability distribution of the t-step random walk starting at v. ($p_{t,v}$ is truncated i.e. low probability entries are set to zero, in order to avoid exploring too much of the graph.) For each t, let π be the permutation on the vertices of the graph that indicates the sorted order of the degree-normalized probabilities i.e.

$$\frac{p_t(\pi(i))}{d(\pi(i))} \geq \frac{p_t(\pi(i+1))}{d(\pi(i+1))} \tag{4.5}$$

The sweep sets $S_1^t, S_2^t, \ldots, S_n^t$ are defined as $S_j^t = \{\pi(1), \ldots, \pi(j)\}$. Let ψ_V be the final stationary distribution of the random walk (all random walks within a component converge to the same stationary distribution.) The main theoretical result exploited says that the difference between $p^t(S_j^t)$ and $\psi_V(S_j^t)$ is either small, or there exists a cut with low conductance among the sweep sets. Therefore by checking the conductance of the sweep sets S_j^t at each time step t, we discover clusters of low conductance.

Andersen and Lang [7] extended this work to handle seed sets (instead of just a seed vertex). On real datasets such as web graph, IMDB graph etc. they select a random subset of nodes belonging to a known community and show that the local clustering approach is able to recover the original community.

Andersen et al. [6] improved upon Spielman and Teng's algorithm by simulating random walks with restarts (i.e. Personalized PageRank), instead of just plain random walks. The notion of sweep sets for probability distributions, obtained by sorting the degree-normalized probabilities, is the same.

The theoretical results here involve pagerank vectors though; if there is a set of vertices whose probability in the pagerank vector is significantly greater than their probability in the general stationary distribution, then some sweep set of the pagerank vector has low conductance. They show that they can compute an approximate page rank vector in time depending only on the error of the approximation and the truncation threshold (and not on the graph size). Once the approximate pagerank vector is computed, conductances of successive sweep sets are calculated to discover a set of vertices with low conductance.

Flow-Based Post-Processing for Improving Community Detection: We will discuss how algorithms for computing the maximum flow in flow networks can be used to post-process or improve existing partitions of the graph. Flake et al. [36] proposed to discover web communities by using a focused crawler to first obtain a coarse or approximate community and then set up a max-flow/min-cut problem whose solution can be used to obtain the actual set of pages that belong to the same community. Lang and Rao [57] discuss a strategy for improving the conductance of any arbitrary bipartition or cut of the graph. Given a cut of the graph $(S, \bar{S})$, their algorithm finds the best improvement among all cuts $(S', \bar{S}')$ such that S' is a strict subset of S. Their main approach is to construct a new instance of a max-flow problem, such that the solution to this problem (which can be found in polynomial time) can be used to find the set S' with the lowest conductance among all subsets of S. They refer to their method as MQI (Max-Flow Quotient-Cut Improvement). They use Metis+MQI to recursively bi-partition the input graph; at each step they bi-partition using Metis first and then improve the partition using MQI and repeat the process on the individual partitions. Anderson and Lang [7] find that MQI can improve the partitions found by local clustering as well.

Community Discovery via Shingling: Broder et al. [19] introduced the idea of clustering web documents through the use of *shingles* and fingerprints (also denoted as *sketches*). In short, a length-s *shingle* is s of all parts of the object. For example, a length-s shingle of a graph node contains s outgoing links of the node; a length-s shingle of a document is a contiguous subsequence of length s of the document. Meanwhile, a *sketch* is a constant-size subset of all shingles with a specific length, with the remarkable property that the similarity between sets of two objects' sketches approximates the similarity between the objects themselves (here the definition of similarity being used is Jaccard similarity, i.e. $sim(A, B) = |A \cap B|/|A \cup B|$. This property makes sketch an object's fingerprint.

Gibson et al. [40] attempt to extract dense communities from large-scale graphs via a recursive application of shingling. In this algorithm, the first-level shingling is performed on each graph node using its outgoing links. That is,

each node v is associated with a sketch of c_1 shingles, each of which stands for s_1 nodes selected from all nodes that v points to. Then an inverted index is built, containing each first-level shingle and a list of all nodes that the shingle is associated with. The second-level shingling function is then performed on each first-level shingle, producing second-level shingles (also called meta-shingles) and sketches. Two first-level shingles are considered as relevant if they share at least one meta-shingle in common, and the interpretation is that these two shingles are associated with some common nodes. If a new graph is constructed in such a way that nodes stand for first-level shingles and edges indicate the above-defined relation, then clusters of first-level shingles correspond to connect components in this new graph. Finally, communities can be extracted by mapping first-level shingles clusters back to original nodes plus including associated common meta-shingles. This algorithm is inherently applicable to both bipartite and directed graph, and can also be extended to the case of undirected graph. It is also very efficient in terms of both memory usage and running time, thus can handle graph of billions of edges.

Alternative Definitions of Communities: At the start of this section, we informally defined a community as a subset of nodes well connected internally and weakly connected to the rest of the graph. We now look at additional notions of communities which are either different from this definition or are refinements of this idea for a particular context.

Asur and Parthasarathy [10] recently introduced the idea of *viewpoint neighborhoods*, which are groups of nodes that are salient or influential from the viewpoint of a single node (or subset of nodes) in the network. Thus a viewpoint neighborhood may be seen as a cluster or community of nodes that is local to the node (or subset of nodes) that is being analyzed. The same paper also proposes algorithms for extraction of viewpoint neighborhoods using activation spread models that are general enough to incorporate different notions of salience or influence. Viewpoint analysis of graphs provides us a novel analytic and conceptual tool for understanding large networks at a fine scale.

Leskovec et al. [60] find that in a wide variety of real-world networks, some of the best communities, according to the measure of conductance (see Equation 4.2) , are groups of nodes that are connected to the rest of the graph by only one edge. They refer to such communities as *whiskers* (with groups of nodes that are connected by 2 edges called *2-whiskers* etc.) They postulate a core-and-whiskers model for the structure of networks, where most networks consist of a core part of the network surrounded by whiskers which are often connected to the rest of the network by only one or two edges. The whiskers of a network may either represent patterns that are useful within the context of the domain or may be considered noise which is to be removed while preprocessing the network.

4. Emerging Fields and Problems

In this section we attempt to identify recent research trends within the domain of community discovery in social networks. Given the relatively preliminary nature of the work presented in this section our objective here is to identify and discuss exemplar efforts rather than provide a comprehensive survey of all results in each sub-area.

4.1 Community Discovery in Dynamic Networks

Most of the community discovery algorithms discussed in Section 3 were designed with the implicit assumption that the underlying network is unchanging. In most real social networks however, the networks themselves as well as the communities and their members evolve over time. Some of the questions consequently raised are: How should community discovery algorithms be modified to take into account the dynamic and temporally evolving nature of social networks? How do communities get formed? How persistent and stable are communities and their members? How do they evolve over time? In this section, we introduce the reader to the slew of recent work that addresses these questions.

Asur et al.[9] presented an an event-based approach for understanding the evolution of communities and their members over time. The key ideas brought forth by this work is a structured way to reason about how communities and individual elements within such networks evolve over time and what are the critical events that characterize their behavior. Events involving communities include *continue, κ-merge, κ-split, form* and *dissolve*, and events involving individuals include *appear, disappear* and *join*. The authors demonstrate how behavioral indices such as stability and influence as well as a diffusion model can be efficiently composed from the events detected by their framework and can be used to effectively analyze real-life evolving networks in an incremental fashion. Their model can also be used to predict future community behavior (e.g. collaboration between groups). Also it can help identifying nodes with interests (e.g. sociable or influential users). Furthermore, semantic content can be integrated in the model naturally.

Recently, much research effort has gone into the question of designing community discovery algorithms for dynamic networks. The simple approach of treating treating each network snapshot as an independent network and applying a conventional community discovery algorithm may result in undesirable fluctuations of community memberships from one snapshot to the next. Consider an extreme example from [25], where there exist two orthogonal splits (A and B) on a data set. A performs slightly better on odd-numbered days, while B is a little superior on even-numbered days. Taking the optimal split every day results in radical change in the obtained communities from day to day,

and therefore it may be better to sacrifice some optimality and instead adopt a consistent split (either A or B) on all days.

Initial approaches to tackle this problem focused on constructing temporal slices of the network and then relied on community discovery on each slice to detect temporal changes to community structure between consecutive slices. For example, Berger-Wolf and Saia[16] took partitions of individuals at each time-stamp as input, trying to find a *metagroup* that contains a sequence of groups which are similar to each other. Definitions of three extreme metagroups (namely most persistent metagroup, most stable metagroup and largest metagroup) were given, and the extraction algorithms were discussed. Tantipathananandh et al. [111] studied the problem of identifying the "true" community affiliations of the individuals in a dynamic network, given the affiliations of the individuals in each timeslice. They formulate this as a combinatorial optimization problem and show that the problem is NP-hard. Consequently they solve the problem using a combination of approximate greedy heuristics and dynamic programming.

An alternative approach to dynamic analysis of networks is to take a holistic view of the community discovery across time-slices, by constraining the network division in a time-slice to not be too divergent from the network divisions of the previous time-slices. Chakrabarti et al.[25] were among the first to work on this problem and referred to it as evolutionary clustering. The most essential contribution of it is, instead of first extracting communities on each network snapshots and then finding connections among communities in different snapshots, it considers *snapshot quality* (how well the clustering at certain time C_t represents the data at t) and *history cost* (how different is the clustering C_t from clustering C_{t-1}) as a whole. In this way, community structure and its evolution are studied at the same time. Furthermore, it allows the compromise between these two parts by linear combination of snapshot quality and history cost. They also adapted agglomerative hierarchical clustering and k-means clustering for this framework.

Sun et al.[106] present an alternative approach to clustering time-evolving graphs using the *Minimum Description Length* (MDL) principle. Here, graphs of consecutive timestamps are grouped into graph stream segments, and these segments are divided by change-points. These change-points indeed indicate points of drastic discontinuities in the network structure. The total cost of graph stream encoding is then defined as $C = \sum_s C^{(s)}$, where $C^{(s)}$ is the encoding cost for s-th graph stream segment. The segment encoding cost, $C^{(s)}$ is again a sum of the segment length, the graph encoding cost and the partition encoding cost. Unfortunately, minimizing total cost was proved NP-hard, leading to a greedy algorithm based on alternating minimizations called GraphScope. Basically it deals with when to start a new graph stream segment and how to

find well-formed communities among all snapshots in a single segment. One of GraphScope's advantages is that it doesn't require any parameter as input.

Chi et al.[26] extended spectral clustering to a dynamic network setting. They proposed two frameworks, named *preserving cluster quality* (PCQ) and *preserving cluster membership* (PCM) respectively, to measure the temporal/history cost. The former metric is interested in how well the partition at time t (C_t) performs on the data at time $t-1$, while latter cares how similar the two consecutive partitions (C_t and C_{t-1}) are. This framework also allows variation in cluster numbers as well as insertion and removal of nodes.

Lin et al.[63] proposed *FacetNet* for dynamic community discovery through the use of probabilistic community membership models. The advantage of such probabilistic models is the ability to assign each individuals to multiple communities with a weight indicating the degree of membership for each community. They used KL-divergence to measure the snapshot quality and history cost respectively. It was proved in[64] that when certain assumptions hold, optimization of total cost is equivalent to maximization of the log-likelihood function $L(U_t) = \log P(W_t|U_t) + \log P(U_t|U_{t-1})$, where W_t is the data at time t, and U_t the cover at t.

Kim and Han[55] revisited the cost function used in existing research and found that temporal smoothing at the clustering level can degrade the performance because of the need to adjust the clustering result iteratively. Their remedy was to *push down* the cost to each pair of nodes, get a temporal-smoothed version of pair-wise node distance and then conduct density-based clustering on this new distance metric. To deal with the problem that the number of communities change over time, greedy local clustering mapping based on mutual information was performed. By doing so, the model can account for the arbitrary creation/dissolution as well as growing/shrinking of a community over time.

4.2 Community Discovery in Heterogeneous Networks

Most conventional algorithms assume the existence of a homogeneous network where nodes and edges are of the same type. In the real world, however, we often have to deal with heterogeneous social networks, where the nodes are of different kinds, edges are of dissimilar types (e.g. relationships based upon various communication methods[43]) or even both of them at the same time[108]. Consider an IMDB network, where the entities may be of multiple types such as movies, directors, actors and the relationships may also be of different types such as acted-in, directed-by, co-acted-in and so on. Such diversity presents both an opportunity and challenge, since there may exist valuable information to be gained from recognizing the heterogeneity in the network and

yet it is not obvious how to appropriately handle nodes and edges that belong to different types.

Guy et al.[43] designed the SONAR API, aiming at aggregating social network information from emails, instant messaging, organization charts, blogs and so on. They experimented with different weighted combination of these information sources and subsequently performed the task of user recommendation based on the outcome network. It was reported that recommendation based on aggregated network had a better performance than that based on any of the input networks. However, they did not discuss how to find the best combination scheme.

Cai et al.[23] looked into the problem of finding the best linear combination of different source networks. Their main idea is to first build a target network, with associated adjacency matrix $\tilde{M}$, and regress it on the source networks M_i.

$$\mathbf{a}^{opt} = \arg\min_{\mathbf{a}} \|\tilde{M} - \sum_{i=1}^{n} a_i M_i\|^2 \tag{4.6}$$

The a_is are the coefficients for the corresponding source networks. However, since we rarely know the full target network, the authors assume that the user provides only a few example target relationships, and derive a linear programming formulation that efficiently solves the linear regression problem.

The NetClus algorithm introduced by Sun et al.[108] dealt with clustering networks with *star network* schema. In the star network, each record is actually a compound of a single *target type* and several *attribute types*. The decision of cluster assignment is made by ranking the posterior probabilities resulting from a generative model. This ranking-assignment process is then iterated until convergence. By taking advantage of the ranking distribution for each type of objects (e.g. conference, author and topic), importance/influence ranking in each type can be retrieved as well as communities themselves. Therefore the results become more meaningful and interpretable. The usage of this algorithm, however, is limited by its ability to process only networks with *star network* schema. Similarly, the RankClus algorithm[107] is only designed to deal with bi-type network, where the network's vertex set have two types of vertices.

Finally, we also mention that ensemble clustering [105, 8] - an approach where the results of multiple clusterings are combined - is also a potential solution for clustering heterogeneous networks.

4.3 Community Discovery in Directed Networks

Community discovery has generally concerned itself with undirected networks; however the networks from a number of important domains are essentially directed, e.g. networks of web pages, research papers and Twitter users.

Simply ignoring directionality when analyzing such networks, as has been implicitly done in many studies, both ignores the additional information in the directionality as well as can lead to false conclusions. For this reason, there has recently been some work on community discovery for directed networks.

Many researchers have extended existing objective functions for community discovery from undirected networks (see Section 3.1) to take into account directionality. Using the random-walks interpretation of Normalized Cuts [67], multiple researchers have defined a directed version of Normalized Cuts. Let P be the transition matrix of a random walk on the directed graph, with π being its associated stationary distribution vector (e.g. PageRank vector) satisfying $\pi P = \pi$. The (directed) Normalized Cut for a group $S \subset V$ is given as [127, 28, 45, 66]:

$$Ncut_{dir}(S) = \frac{\sum_{i \in S, j \in \bar{S}} \pi(i) P(i,j)}{\sum_{i \in S} \pi(i)} + \frac{\sum_{j \in \bar{S}, i \in S} \pi(j) P(j,i)}{\sum_{j \in \bar{S}} \pi(j)} \tag{4.7}$$

The above objective function is often minimized using spectral clustering - this time by post-processing the top eigenvectors of the directed Laplacian, defined as [127, 28, 45, 66] (P and Π are defined as above):

$$\mathcal{L} = I - \frac{\Pi^{1/2} P \Pi^{-1/2} + \Pi^{-1/2} P' \Pi^{1/2}}{2} \tag{4.8}$$

Leicht and Newman[58] introduced the directed version of modularity[78] as follows:

$$Q = \frac{1}{m} \sum_{ij} [A_{ij} - \frac{k_i^{in} k_j^{out}}{m}] \delta_{c_i, c_j} \tag{4.9}$$

where k_i^{in} is the in-degree of node i, and k_j^{out} the out-degree of j. To fit the new metric into spectral optimization method proposed in[77] where a large community is bisected at each step, the definition of modularity matrix $\mathbf{B}$ is modified as $B_{ij} = A_{ij} - \frac{k_i^{in} k_j^{out}}{m}$. Furthermore, the modularity function is rewritten as

$$Q = \frac{1}{4m} \mathbf{s}^T (\mathbf{B} + \mathbf{B}^T) s \tag{4.10}$$

since $\mathbf{B}$ alone may not be symmetric. However, the algorithm may still suffer from the resolution problem, as pointed out by Fortunato and Barthélemy[38].

Satuluri and Parthasarathy [94] argue that a clustering with low directed normalized cut or high directed modularity is often not the most meaningful way to cluster directed graphs. In particular, such objective functions still favor clusters with high inter-connectivity, while inter-connectivity is not necessary for a group of vertices to form a meaningful cluster in directed networks [94]. They argue instead for a more general framework where we first convert the

input directed graph into a weighted, undirected graph using a (symmetric) similarity measure for the vertices of the directed graph. They find that a similarity measure that uses in-link and out-link similarity while also discounting common links to highly connected nodes is more effective than existing approaches at discovering communities from directed networks.

4.4 Coupling Content and Relationship Information for Community Discovery

Although relationship information of social networks has been extensively investigated, the work of incorporating content and relationship information to facilitate community discovery has not been thoroughly studied yet. In fact this problem is at the heart of recent efforts to analyze social media. Relationship information can be viewed as a plain graph with vertices and edges, while content information are properties attached to these graph elements. Content may exist in the form of text, images, or even geographical locations. With the availability of content information, it is expected that the extracted communities are not only topologically well-connected, but also semantically coherent and meaningful. Consider the email communication network where sender-recipient communication can be modeled as user relationship. Then a spammer account will have a large amount of connections with others and thus be regarded as the center of a new community, which is useless in most cases. The importance of utilizing content information can be clearly perceived from this simple example. Although in previous studies many datasets also contain rich contents, they are merely used to infer user relationships (e.g. establish a link between two authors of a research paper), not to contribute to community extraction.

Content information may be in the forms of user profile or user-created material, in which case they are associated with vertices. Content may also be associated with edges in the network, as we will see in some literatures discusses below. In some cases, it's more intuitive to use "attribute" instead of "content", thus they are used interchangeably in the following context. The problem of interest is: how can communities be found, using both relationship and content information?

First introduced are three approaches using Bayesian generative models, aiming to incorporating textual contents. The Group-Topic model proposed by Wang et al.[120] is an extension of the stochastic blockstructures model [79], where both relations and their attributes are considered. Here, an entity is related with another if they behave the same way on an event, and texts associated with the event are this relationship's attributes. Furthermore, each event corresponds to one of the T latent topics. Therefore, the group membership of an entity is no longer constant, but changes over different topics. This blueprint

of directed probabilistic model guides the discovery of groups by topics, and vice-versa.

Zhou et al.[128] introduce the notion of a *semantic community* and two corresponding Community-User-Topic (CUT) models. Their objective is to extract semantic communities from communication documents. In the CUT_1 model, the distribution of topics is conditioned on users, who are, in turn, conditioned on communities. This algorithm is similar to conventional community discovery algorithms, in the sense that a community is still defined as no more than a group of users. On the contrary, the CUT_2 model let communities decide topics and topics decide users, assuming a tighter connection between community and topic. As the experiments report, CUT_2 model finds higher-quality semantic communities, and is computationally more efficient than CUT_1.

Pathak et al.[84] presented the Community-Author-Recipient-Topic (CART) model in a setting of email communication networks, assuming that the discussion among users within a community is relevant to these users as well as the community. The model constrains all users involved and topics discussed in the email conversation to belong to a single community, while same users and topics in different conversations can be assigned to different communities. Compared with previous models including CUT models, this model is argued to emphasize more on how topics and relationships *jointly* affect community structure. Yet, all these three methods suffer from a common disadvantage: inference of the generative model using Gibbs sampling may converge slowly, thus the running time may be a problem in practice, especially for large-scale datasets.

The problem of Connected *X* Clusters (*CXC*) introduced by Moser et al.[70] was inspired by traditional graph clustering. While the algorithm still assumes that each cluster is compact and distinctive from neighboring ones (by using content information), the idea of *community* is enforced by requiring each cluster to be *internally connected* (by using relationship information). They also formally derived the number of initial centroids such that each true cluster is represented by at least one initial cluster atom (the smallest building component in the algorithm), at certain pre-defined confidence level. The proposed algorithm (called *JointClust*) is essentially a agglomerative clustering method. It first determines cluster atoms based on the number of initial centroids. In the second phase, it merges cluster atoms in a bottom-up fashion based on the *joint Silhouette coefficient*, an extension of traditional Silhouette coefficient [52]. This algorithm does not require pre-specified cluster number. It, however, still takes as a parameter the minimum size of each cluster.

Negoescu and Gatica-Perez[73] proposed an algorithm to identify communities of groups on Flickr, an image-sharing website. In the context of this algorithm, groups refer to self-organized sets of Flickr users, and are the elements of the final communities that we are looking for. Therefore, a community is

also referred to as a *hypergroup*. The procedure is to first abstract each group into a bag-of-tags model, where tags come from the group's images and can be regarded as contents generated by the group. Then latent Dirichlet allocation *LDA* method is applied, giving distribution of latent topics over each group. Several similarity measures can be exploited to build the similarity matrix for groups, and the original problem is cast to a clustering problem on similarity matrix. Although it's not discussed in the paper, this algorithm is applicable to finding communities of users. Again, efficiency may be a potential concern, which is intrinsic to all latent-topic-based approaches.

5. Crosscutting Issues and Concluding Remarks

In this article we surveyed the principal results on community discovery within social networks. We first examined the contexts and use-case scenarios for community discovery within various social settings. We next took a look at the core methods developed to extract community structure from large networks ranging from the traditional to the current state-of-the-art. Subsequently we focused on recent and emerging strands of research that is gaining traction within this community. Below we briefly highlight four cross-cutting research themes that are likely to play a significant role as we move forward within this field. Note, that this is by no means a comprehensive list of cross-cutting issues but highlight some of the key challenges and opportunities within the field.

- Scalable Algorithms: With the size and scale of networks and information involved researchers are increasingly turning to scalable, parallel and distributed algorithms for community discovery within social networks. At the algorithmic level multi-level algorithms relying on graph coarsening and refinement offer potential [51, 31, 93]. Architecture conscious algorithms on the GPU and multi-cores offer another orthogonal approach [22, 83] as do streaming algorithms[3]. Given the recent trend towards cloud computing, researchers are beginning to investigate algorithms for community discovery on platforms such as Hadoop [81, 48, 49].

- Visualization of Communities and their Evolution: Visualizing large complex networks and honing in on important topological characteristics is a grand challenge since one often runs out of pixel space especially when attempting to characterize the behavior of billion node networks. This area, particularly in the context of community discovery within social networks has seen limited research thus far [119, 125, 21, 85]. Moving forward we envision multiple roles for visualization in this context. First as a front end to display dynamic (sub-)networks (details-on-demand) housed within the warehouse (e.g. visualizing a trust network). Second, as a mechanism to help understand and drive

the analysis process. Third, as a means to validate and lend transparency to the discovery process. An important challenge here is to determine how dynamic information is to be modeled and visualized effectively and efficiently.

- Incorporating Domain Knowledge:It has been our observation that often we as data mining researchers tend to under-utilize available domain information during the pattern discovery or model building process. In fact data mining researchers often specifically omit important domain knowledge from the training phase as it then allows them a means to independently confirm the utility of the proposed methods during validation and testing. While useful, such a methodology often limits scientific advances within the domain. We believe a fresh look at how domain knowledge can be embedded in existing approaches and better testing and validation methodologies in close conjunction with domain experts must be designed (see for example work in the field of Bioinformatics). It is our hypothesis that domain knowledge is often too valuable a resource to simply ignore during the discovery phase as it can be an effective means to prune and guide the search for interesting patterns.

- Ranking and Summarization: While ranking and summarizing patterns has been the subject of much research in the data mining community the role of such methods in this community has been much less researched. As networks become larger and particularly with an increasing focus on dynamic networks identifying a hierarchy of patterns from most important to least important becomes crucial to help domain experts focus on the key insights to be drawn from the analysis. Leveraging domain information (as noted above) will be crucial for this endeavor.

In conclusion we would like to add that the field of community discovery in networks (particularly social) is still fairly new with a number of open and exciting problems ranging from the theoretical to the empirical and covering a gamut of core research areas both within computer science and across disciplines. Given the dynamic nature of the field and the broad interest across multiple disciplines we expect to see many more exciting results on this topic in the future.

5.0.1 Acknowledgments. This work is supported in part by the following grants: NSF CAREER IIS-0347662, CCF-0702587, IIS-0917070. The authors would also like to thank Charu Aggarwal and the anonymous reviewers for useful comments for improving this article.

References

[1] J. Abello, P. Pardalos, and MGC Resende. On maximum clique problems in very large graphs. In *External memory algorithms*, pages 119–130. American Mathematical Society, 1999.

[2] L.A. Adamic and N. Glance. The political blogosphere and the 2004 US election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*, page 43. ACM, 2005.

[3] Charu C. Aggarwal, Yuchen Zhao, and Philip S. Yu. On clustering graph streams. In *SDM*, pages 478–489, 2010.

[4] R. Albert, H. Jeong, and A.L. Barabási. Diameter of the World-Wide Web. *Nature*, 401(6749):130–131, 1999.

[5] P. Aloy and R.B. Russell. The third dimension for protein interactions and complexes. *Trends in biochemical sciences*, 27(12):633–638, 2002.

[6] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 475–486, Washington, DC, USA, 2006. IEEE Computer Society.

[7] R. Andersen and K.J. Lang. Communities from seed sets. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, page 232. ACM, 2006.

[8] S. Asur, S. Parthasarathy, and D. Ucar. An ensemble approach for clustering scalefree graphs. In *LinkKDD workshop*, 2006.

[9] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 913–921, New York, NY, USA, 2007. ACM.

[10] Sitaram Asur and Srinivasan Parthasarathy. A viewpoint-based approach for interaction graph analysis. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 79–88, New York, NY, USA, 2009. ACM.

[11] A.L. Barabási and E. Bonabeau. Scale-free networks. *Scientific American*, 288(5):60, 2003.

[12] A.L. Barabási and RE Crandall. Linked: The new science of networks. *American journal of Physics*, 71:409, 2003.

[13] S.T. Barnard and H.D. Simon. Fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. *Concurrency Practice and Experience*, 6(2):101–118, 1994.

[14] J. Bascompte, P. Jordano, C.J. Melián, and J.M. Olesen. The nested assembly of plant–animal mutualistic networks. *Proceedings of the National Academy of Sciences of the United States of America*, 100(16):9383, 2003.

[15] M.L. Bech and E. Atalay. The topology of the federal funds market. *Working Paper Series*, 2008.

[16] T.Y. Berger-Wolf and J. Saia. A framework for analysis of dynamic social networks. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 528. ACM, 2006.

[17] Board on Army Science and Technology. *Strategy for an Army Center for Network Science, Technology, and Experim entation*. The National Academies Press, 2007.

[18] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On finding graph clusterings with maximum modularity. In *Graph-Theoretic Concepts in Computer Science*, pages 121–132. Springer, 2007.

[19] A.Z. Broder, S.C. Glassman, M.S. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.

[20] S. Brohee and J. Van Helden. Evaluation of clustering algorithms for protein-protein interaction networks. *BMC bioinformatics*, 7(1):488, 2006.

[21] G. Buehrer and K. Chellapilla. A scalable pattern mining approach to web graph compression with communities. In *WSDM '08: Proceedings of the international conference on Web search and web data mining*, pages 95–106, New York, NY, USA, 2008. ACM.

[22] G. Buehrer, S. Parthasarathy, and M. Goyder. Data mining on the cell broadband engine. In *Proceedings of the 22nd annual international conference on Supercomputing*, pages 26–35. ACM, 2008.

[23] D. Cai, Z. Shao, X. He, X. Yan, and J. Han. Mining hidden community in heterogeneous social networks. In *Proceedings of the 3rd international workshop on Link discovery*, page 65. ACM, 2005.

[24] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1):2, 2006.

[25] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM New York, NY, USA, 2006.

[26] Y. Chi, X. Song, K. Hino, and B.L. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness, October 18 2007. US Patent App. 11/874,395.

[27] F. Chung. Spectral graph theory. *CBMS Regional Conference Series in Mathematics*, 1997.

[28] F. Chung. Laplacians and the Cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, 2005.

[29] A. Clauset, M.E.J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):66111, 2004.

[30] R. DerSimonian and N. Laird. Meta-analysis in clinical trials* 1. *Controlled clinical trials*, 7(3):177–188, 1986.

[31] I.S. Dhillon, Y. Guan, and B. Kulis. Weighted Graph Cuts without Eigenvectors: A Multilevel Approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(11):1944–1957, 2007.

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

[33] J.A. Dunne, R.J. Williams, and N.D. Martinez. Network structure and biodiversity loss in food webs: robustness increases with connectance. *Ecology Letters*, 5(4):558–567, 2002.

[34] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, page 262. ACM, 1999.

[35] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2):298–305, 1973.

[36] G.W. Flake, S. Lawrence, and C.L. Giles. Efficient identification of web communities. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, page 160. ACM, 2000.

[37] G.W. Flake, S. Lawrence, C.L. Giles, and F.M. Coetzee. Self-organization of the web and identification of communities. *Communities*, 35(3):66–71, 2002.

[38] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36, 2007.

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

[40] D. Gibson, R. Kumar, and A. Tomkins. Discovering Large Dense Subgraphs in Massive Graphs. In *VLDB '05: Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, August 30-September 2, 2005*, page 721. ACM, 2005.

[41] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821, 2002.

[42] W. Glanzel and A. Schubert. Analysing scientific networks through coauthorship. *Handbook of quantitative science and technology research*, pages 257–276, 2004.

[43] I. Guy, M. Jacovi, E. Shahar, N. Meshulam, V. Soroka, and S. Farrell. Harvesting with SONAR: the value of aggregating social network information. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1017–1026. ACM, 2008.

[44] G. C. Homans. *The Human Group*. New York: Harcourt, Brace, 1950.

[45] J. Huang, T. Zhu, and D. Schuurmans. Web communities identification from random walks. *Lecture Notes in Computer Science*, 4213:187, 2006.

[46] R.F. i Cancho. The small world of human language. *Proceedings of the Royal Society B: Biological Sciences*, 268(1482):2261–2265, 2001.

[47] H. Jeong, S.P. Mason, A.L. Barabási, and Z.N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, 2001.

[48] U. Kang, C. Tsourakakis, A.P. Appel, C. Faloutsos, and J. Leskovec. Radius plots for mining tera-byte scale graphs: Algorithms, patterns, and observations. In *SIAM International Conference on Data Mining*, 2010.

[49] U Kang, C.E Tsourakakis, and C. Faloutsos. Pegasus: Mining peta-scale graphs. *Knowledge and Information Systems*, 2010.

[50] R. Kannan, S. Vempala, and A. Veta. On clusterings-good, bad and spectral. In *FOCS '00*, page 367. IEEE Computer Society, 2000.

[51] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20, 1999.

[52] L. Kaufman and PJ Rousseeuw. Finding groups in data; an introduction to cluster analysis. *Wiley Series in Probability and Mathematical Statistics. Applied Probability and Statistics Section (EUA).*, 1990.

[53] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146, New York, NY, USA, 2003. ACM.

[54] B. Kernighan and S. Lin. An Efficient Heuristic Procedure for partitioning graphs. *The Bell System Technical J.*, 49, 1970.

[55] M.S. Kim and J. Han. A particle-and-density based evolutionary clustering method for dynamic networks. *Proceedings of the VLDB Endowment*, 2(1):622–633, 2009.

[56] Y. Koren. The BellKor Solution to the Netflix Grand Prize. *KorBell Team's Report to Netflix*, 2009.

[57] K. Lang and S. Rao. A flow-based method for improving the expansion or conductance of graph cuts. *Lecture notes in computer science*, pages 325–337, 2004.

[58] E.A. Leicht and M.E.J. Newman. Community structure in directed networks. *Physical review letters*, 100(11):118703, 2008.

[59] J. Leskovec, L.A. Adamic, and B.A. Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5, 2007.

[60] J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *CoRR*, abs/0810.1355, 2008.

[61] J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW '08*, pages 695–704, New York, NY, USA, 2008. ACM.

[62] L. Li, C.J. Stoeckert, and D.S. Roos. OrthoMCL: identification of ortholog groups for eukaryotic genomes. *Genome Res*, 13(9):2178–2189, September 2003.

[63] Y.R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B.L. Tseng. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 685–694, New York, NY, USA, 2008. ACM.

[64] Y.R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B.L. Tseng. Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(2):1–31, 2009.

[65] D. Lusseau. The emergent properties of a dolphin social network. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 270(Suppl 2):S186, 2003.

[66] M. Meila and W. Pentney. Clustering by weighted cuts in directed graphs. In *Proceedings of the 7th SIAM International Conference on Data Mining*, pages 135–144. Citeseer, 2007.

[67] M. Meila and J. Shi. A random walks view of spectral segmentation. *AI and Statistics (AISTATS)*, 2001, 2001.

[68] J. Memmott, N.M. Waser, and M.V. Price. Tolerance of pollination networks to species extinctions. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 271(1557):2605, 2004.

[69] J.M. Montoya et al. Small world patterns in food webs. *Journal of theoretical biology*, 214(3):405–412, 2002.

[70] F. Moser, R. Ge, and M. Ester. Joint cluster analysis of attribute and relationship data withouta-priori specification of the number of clusters. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 510–519. ACM New York, NY, USA, 2007.

[71] F. Murray. Innovation as co-evolution of scientific and technological networks: exploring tissue engineering. *Research Policy*, 31(8-9):1389–1403, 2002.

[72] S.F. Nadel. The Theory of Social Structure. *London: Cohen and West*, 1957.

[73] R.A. Negoescu, B. Adams, D. Phung, S. Venkatesh, and D. Gatica-Perez. Flickr hypergroups. In *Proceedings of the seventeen ACM international conference on Multimedia*, pages 813–816. ACM, 2009.

[74] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, 2004.

[75] M.E.J. Newman. Assortative mixing in networks. *Physical Review Letters*, 89(20):208701, 2002.

[76] M.E.J. Newman. A measure of betweenness centrality based on random walks. *Social networks*, 27(1):39–54, 2005.

[77] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577, 2006.

[78] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb 2004.

[79] K. Nowicki and T.A.B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

[80] L. Palen and S.B. Liu. Citizen communications in crisis: anticipating a future of ICT-supported public participation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, page 736. ACM, 2007.

[81] S. Papadimitriou and J. Sun. Disco: Distributed co-clustering with Map-Reduce: A case study towards petabyte-scale end-to-end mining. In *Eighth IEEE International Conference on Data Mining, 2008. ICDM'08*, pages 512–521, 2008.

[82] S. Parthasarathy. Data mining at the crossroads: successes, failures and learning from them. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 1055. ACM, 2007.

[83] S. Parthasarathy, S. Tatikonda, G. Buehrer, and A. Ghoting. *Next Generation of Data Mining*, chapter Architecture Conscious Data Mining: Current Directions and Future Outlook, pages 261–280. Chapman and Hall/CRC, 2008.

[84] N. Pathak, C. DeLong, A. Banerjee, and K. Erickson. Social topic models for community extraction. In *The 2nd SNA-KDD Workshop*, volume 8, 2008.

[85] A. Perer and B. Shneiderman. Balancing systematic and flexible exploration of social networks. *IEEE Transactions on Visualization and Computer Graphics*, pages 693–700, 2006.

[86] J. Podani, Z.N. Oltvai, H. Jeong, B. Tombor, A.L. Barabási, and E. Szathmary. Comparable system-level organization of Archaea and Eukaryotes. *Nature genetics*, 29(1):54–56, 2001.

[87] P. Raghavan. Social networks: from the Web to the enterprise. *IEEE Internet Computing*, 6(1):91–94, 2002.

[88] P.K. Reddy, M. Kitsuregawa, P. Sreekanth, and S.S. Rao. A graph based approach to extract a neighborhood customer community for collaborative filtering. In *Databases in networked information systems: second international workshop, DNIS 2002, Aizu, Japan, December 16-18, 2002: proceedings*, page 188. Springer-Verlag New York Inc, 2002.

[89] S.A. Rice. The identification of blocs in small political bodies. *The American Political Science Review*, 21(3):619–627, 1927.

[90] Y. Richter, E. Yom-Tov, and N. Slonim. Predicting customer churn in mobile networks through analysis of social groups. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, 2010.

[91] E.M. Rogers. *Diffusion of innovations*. Free Pr, 1995.

[92] E.M. Rogers and D.L. Kincaid. *Communication networks: Toward a new paradigm for research*. Free Pr, 1981.

[93] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *KDD '09*, pages 737–746, New York, NY, USA, 2009. ACM.

[94] V. Satuluri and S. Parthasarathy. Symmetrizations for clustering directed graphs. In *Workshop on Mining and Learning with Graphs, MLG 2010*, 2010. Also available as technical report from `ftp://ftp.cse.ohio-state.edu/pub/tech-report/2010/TR12.pdf`.

[95] V. Satuluri, S. Parthasarathy, and D. Ucar. Markov Clustering of Protein Interaction Networks with Improved Balance and Scalability. In *Proceedings of the ACM Conference on Bioinformatics and Computational Biology*, 2010.

[96] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[97] R.V. Solé and M. Montoya. Complexity and fragility in ecological networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1480):2039, 2001.

[98] R.V. Solé and R. Pastor-Satorras. Complex networks in genomics and proteomics. *Handbook of Graphs and Networks*, pages 147–169, 2002.

[99] E.D. Sontag. Structure and stability of certain chemical networks andapplications to the kinetic proofreading model of T-cell receptor signaltransduction. *IEEE transactions on automatic control*, 46(7):1028–1047, 2001.

[100] D.A. Spielman and N. Srivastava. Graph sparsification by effective resistances. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 563–568, New York, NY, USA, 2008. ACM.

[101] D.A. Spielman and S.H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 81–90. ACM New York, NY, USA, 2004.

[102] E. Sprinzak, S. Sattath, and H. Margalit. How reliable are experimental protein-protein interaction data? *Journal of molecular biology*, 327(5):919–923, 2003.

[103] J. Srivastava, R. Cooley, M. Deshpande, and P.N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *ACM SIGKDD Explorations Newsletter*, 1(2):23, 2000.

[104] M. Steenstrup. Cluster-based networks. In *Ad hoc networking*, page 138. Addison-Wesley Longman Publishing Co., Inc., 2001.

[105] A. Strehl and J. Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[106] J. Sun, C. Faloutsos, S. Papadimitriou, and P.S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 687–696. ACM New York, NY, USA, 2007.

[107] Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, and T. Wu. RankClus: integrating clustering with ranking for heterogeneous information network analysis. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 565–576. ACM, 2009.

[108] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 797–806. ACM, 2009.

[109] S.R. Sundaresan, I.R. Fischhoff, J. Dushoff, and D.I. Rubenstein. Network metrics reveal differences in social organization between two fission–fusion species, Grevy's zebra and onager. *Oecologia*, 151(1):140–149, 2007.

[110] P.N. Tan, M. Steinbach, and V. Kumar. *Introduction to data mining*. Pearson Addison Wesley Boston, 2006.

[111] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 726. ACM, 2007.

[112] S.H. Teng. Coarsening, sampling, and smoothing: Elements of the multilevel method. *Algorithms for Parallel Processing*, 105:247–276, 1999.

[113] N.M. Tichy, M.L. Tushman, and C. Fombrun. Social network analysis for organizations. *Academy of Management Review*, 4(4):507–519, 1979.

[114] P. Uetz, L. Giot, G. Cagney, T.A. Mansfield, R.S. Judson, J.R. Knight, V. Lockshon, D. a nd Narayan, M. Srinivasan, P. Pochart, et al. A comprehensive analysis of protein–protein interactions in Saccharomyces cerevisiae. *Nature*, 403(6770):623–627, 2000.

[115] S. Van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, 2000.

[116] F.A. Von Hayek. The use of knowledge in society. *American Economic Review*, 35(4):519–530, 1945.

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

[118] A. Wagner and D.A. Fell. The small world inside large metabolic networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 268(1478):1803, 2001.

[119] N. Wang, S. Parthasarathy, K.L. Tan, and A.K.H. Tung. CSV: visualizing and mining cohesive subgraphs. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 445–458. ACM, 2008.

[120] X. Wang, N. Mohanty, and A. McCallum. Group and topic discovery from relations and their attributes. *Advances in Neural Information Processing Systems*, 18:1449, 2006.

[121] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*. Cambridge Univ Pr, 1994.

[122] D.J. Watts. *Small worlds: the dynamics of networks between order and randomness*. Princeton Univ Press, 2003.

[123] R.S. Weiss and E. Jacobson. A method for the analysis of the structure of complex organizations. *American Sociological Review*, 20(6):661–668, 1955.

[124] J. Xu and H. Chen. Criminal network analysis and visualization. *Commun. ACM*, 48(6):100–107, 2005.

[125] X. Yang, S. Asur, S. Parthasarathy, and S. Mehta. A visual-analytic toolkit for dynamic interaction graphs. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1016–1024. ACM, 2008.

[126] W.W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977.

[127] D. Zhou, J. Huang, and B. Scholkopf. Learning from labeled and unlabeled data on a directed graph. In *ICML '05*, pages 1036–1043, 2005.

[128] D. Zhou, E. Manavoglu, J. Li, C.L. Giles, and H. Zha. Probabilistic models for discovering e-communities. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, page 182. ACM, 2006.