Partitioning Networks with Node Attributes by Compressing Information Flow

Laura M. Smith
Department of Mathematics
California State University
Fullerton, CA
lausmith@fullerton.edu

Linhong Zhu
Information Sciences Institute
U. of Southern California
Marina del Rey, CA 90292
linhong@isi.edu

Kristina Lerman
Information Sciences Institute
U. of Southern California
Marina del Rey, CA 90292
lerman@isi.edu

Allon G. Percus
Claremont Graduate U.
Claremont, CA 91711
allon.percus@cgu.edu

ABSTRACT

Real-world networks are often organized as modules or communities of similar nodes that serve as functional units. These networks are also rich in content, with nodes having distinguishing features or attributes. In order to discover a network's modular structure, it is necessary to take into account not only its links but also node attributes. We describe an information-theoretic method that identifies modules by compressing descriptions of information flow on a network. Our formulation introduces node content into the description of information flow, which we then minimize to discover groups of nodes with similar attributes that also tend to trap the flow of information. The method has several advantages: it is conceptually simple and does not require ad-hoc parameters to specify the number of modules or to control the relative contribution of links and node attributes to network structure. We apply the proposed method to partition real-world networks with known community structure. We demonstrate that adding node attributes helps recover the underlying community structure in content-rich networks more effectively than using links alone. In addition, we show that our method is faster and more accurate than alternative state-of-the-art algorithms.

1. INTRODUCTION

One of the fundamental tasks in network analysis is to partition a network into clusters, or modules, of similar nodes, which often correspond to functional units in biological networks [21, 22] or communities in social networks [19]. The vast majority of methods developed for this task rely on network topology, i.e., the structure of links between nodes, and treat the nodes themselves as indistinguishable. For example, spectral partitioning methods [5, 30, 28] identify which links to cut to separate the network into disconnected components, while modularity-based approaches [19, 8] find clusters of densely connected nodes. Real-world networks, however, are often rich in content, with nodes that have distinguishing features or attributes. Individuals in a social network differ in age, gender, education and interests, while articles in a scientific paper citation network have different words and topics. The similarities and differences in the content of nodes can affect the patterns of linking, particularly in social networks [18, 14], and taking them into account may improve the quality of the discovered modules. This observation has inspired several attempts to partition content-rich networks [20, 32, 24, 34, 35, 36, 37]. In contrast to these works, we describe a parameter-free, conceptually simple method that combines information in links and node attributes to partition a network.

Our method is situated in the information-theoretic framework introduced by Rosvall & Bergstrom [23] for finding the modular structure of networks. Their approach is inspired by an observation that information flows on a network tend to get trapped within modules. As a consequence, it is possible to compress the description of information flow by reusing names of nodes in different modules. Using random walks as a proxy for information flow, their method partitions the network so as to minimize the Map Equation, which gives the expected description length of a random walk. Thus, the approach exploits the duality between identifying structure and the compression problem to identify the optimal number of modules in the network and to assign the nodes to modules.

To describe the flow of information in a content-rich network, however, it is not sufficient to account for the node names and modules. We need an effective means of accounting for node attributes as well. To this end, we introduce the *Content Map Equation*, which incorporates node attributes into a description of information flow, and use it to compress the flow of information on content-rich networks. The Content Map Equation groups nodes into modules not only when information frequently flows between them, but also when they have similar attributes.

Our method has several desirable properties. First, it is conceptually simple and treats links and attributes on an equal footing. It is parameter-free and does not require us to specify the number of modules ahead of time. It is not sensitive to content representation, i.e., how many attributes are used to characterize nodes. Additionally, it does not require a parameter to control the relative contributions of links and attributes in encoding network information. This is contrast to other methods [20, 24, 32, 35, 37], whose quality relies on successfully tuning such a parameter.

Finding a minimum solution to the Content Map Equation is in most cases a hard optimization problem. Similarly to Rosvall & Bergstrom, we use a greedy bottom-up search to find a locally optimal solution. In that procedure, each node starts in its own module, and the search proceeds by merging modules so as to minimize the total description length. However, this becomes intractable for large networks. To address this problem, we propose a top-down search strategy that has better scaling properties than the original greedy algorithm. We show that it leads to dramatically better computational performance without sacrificing result quality.

We use the proposed method to partition several real-world networks with node attributes and a known community structure. We demonstrate that the Content Map Equation identifies better modules than the original Map Equation, which does not use content information. We also show that our method outperforms alternative methods that use both links and attributes, both in terms of runtime and in terms of the quality of the discovered modules.

In the rest of the paper, we first review related work (Section 2), including Rosvall & Bergstrom's Map Equation. In Section 3 we introduce the Content Map Equation that includes node attributes in a description of information in a network. We illustrate on toy networks the difference in the resulting partitions. In Section 4, we describe a greedy bottom-up algorithm that uses the Content Map Equation to minimize the description length of a random walk. The bottom-up algorithm does not scale to large networks; therefore, we propose a top-down algorithm with random restarts that significantly speeds up the compression problem. In Section 5 we use the proposed methods to partition real-world networks with known community structure and demonstrate that our algorithm is faster and more accurate than competing methods.

2. BACKGROUND AND RELATED WORK

Recently, there has been an explosion of interest in community detection using both links and node attributes. Proposed techniques range from generative modeling, to matrix factorization, to information theoretic approaches. Our (non-exhaustive) summary is shown in Table 1.

Most of the existing generative modeling approaches, such as [34, 10, 32, 37], extend the mixed-membership model [9] with the assumption that communities and attributes together generate links. In contrast, Yang et al. [33] assume that communities "generate" both links and attributes, and propose an alternative way to combine content information using probabilistic modeling. However, in practice this approach only supports nodes with single-dimensional attributes due to the embedded logistic modeling, while all remaining approaches using generative modeling in Table 1 support multi-dimensional attributes.

Another popular category for community detection methods using both links and content is the hybrid approach [24, 35, 36]. The general workflow of the hybrid approach is as follows: it first generates content links based on attribute vector similarity, and then combines content links with topological links to detect communities.

Compared to generative modeling and hybrid approaches, fewer methods have been developed that use matrix factorization or information theory. Matrix factorization [15, 20] aims to jointly co-factorize the adjacency matrix of the graph and the node-attribute matrix to obtain the low-ranked node-community matrix.

From the information theoretic view, the entropy-based approach [6] aims to detect communities with low entropy and high modularity. Akoglu [1] extracts cohesive subgraphs by compressing the storage cost of matrices.

In our work, we approach the problem of partitioning content-rich networks from another information theoretic perspective: exploiting the duality between identifying communities and compressing information, which differs from the matrix storage compression [1]. Our method is inspired by Rosvall & Bergstrom [23], who proposed compressing information flows on a network in order to identify modules. Using random walks as a proxy for information flow, their method compresses the description length of a random walk by minimizing the *Map Equation*. Through this optimization, communities emerge as modules with large internal information flows form. We adopt a similar approach but incorporate content, with information from links and node attributes contributing equally to module discovery. Below, we briefly describe the Map Equation method.

2.1 Compressing Random Walks

We first need a method to encode the path traversed by a random walk on a network. Consider the set of nodes, and assign to each node a codeword, such as a Huffman code [12]. Huffman encoding gives more frequent codewords a shorter length, whereas less common codewords get longer description lengths. The length of a codeword is taken to be the number of bits required to represent it. We expect the nodes that are visited more often by a random walk to have shorter codeword lengths.

Consider X, a random variable with n possible states, where the ith state occurs with frequency x_i. Then according to Shannon's source coding theorem [26], in order to describe the n codewords representing the possible states, the average codeword length must be greater than or equal to the entropy

$$H(X) = -\sum_{i=1}^{n} x_i \log_2(x_i).$$

This is the basis for the Map Equation, which aims to minimize the full description length of a code based on the average codeword length.

The network description length is calculated at two levels, the node level and the module level. A module is a group of nodes that have been merged, i.e., a community. Without any modules, n distinct codewords are required to represent the n nodes. The more nodes, the longer the longest codeword will be. Consider a partition of the nodes into m modules. For each module, there is a set of codewords to represent the nodes within the module, which can be reused in other modules. This shortens the length of the longest codeword that describes the nodes.

While the longest codeword for nodes is shorter, the description must now also take into account codewords to represent which module was entered by the path. It may seem counterproductive to have two codewords to locate a single node. However, when describing a path on a network, if the random walker remains in a particular module for a long time before switching modules, then the codewords for indicating module entrance are used less frequently. Therefore, merging nodes into modules is advantageous when the nodes

Table 1: Summary of related work on community detection using both links and node attributes. n: number of nodes, l: number of links, d: number of attributes, k: number of iterations, m: number of communities, and $\delta_n/\delta_l/\delta_d$: number of nodes/links/attributes in the neighborhood of a node. MF and IR stand for matrix factorization and information theory-based approaches. The last two columns list features of the methods proposed in this paper.

Method	*Generative*					*Hybrid*			*MF*		*IR*		*Proposed*	
	[10]	[32]	[33]	[34]	[37]	[24]	[35]	[36]	[15]	[20]	[1]	[6]	B-CME	T-CME
Multi-attribute	√		√		√	√		√	√	√	√	√	√	√
Parameter-free								√		√	√	√	√	√
$O(km^2(n+l+d))$ and above	√	?		√		?	√		√	?	√	?	√	
$O(kmn(\delta_n+\delta_l+\delta_d))$			√		√			√						√

form a dense cluster with few links to other modules. If this is the case, then the modules form communities where the information flow is greater within the module, highlighting the relationship of the nodes.

2.2 The Map Equation

The Map Equation [23] gives the average description length for a step of an infinite random walk. We now review the details for the Map Equation, a two-level description of the network. At the first level, modules are connected to other modules. At the second level, nodes are connected to others within the module. Thus, we need to incorporate codewords at both levels into the Map Equation.

Given partition M of n nodes into m modules, let $q_{i\curvearrowright}$ be the probability of exiting module i. Then $q_{\curvearrowright} = \sum_{i=1}^{m} q_{i\curvearrowright}$ gives the probability that the random walk leaves a module in a given step. From this, we find the entropy of the movement between the modules,

$$H(\mathcal{Q}) = -\sum_{i=1}^{m} \frac{q_{i\curvearrowright}}{q_{\curvearrowright}} \log_2\left(\frac{q_{i\curvearrowright}}{q_{\curvearrowright}}\right). \tag{1}$$

This average codeword length is then weighted by the frequency with which a path exits a module, giving the first term in the Map Equation, $q_{\curvearrowright}H(\mathcal{Q})$.

For the second term of the Map Equation, we look within each module and examine the possible steps for a random walker. A random walker can either move to another node within the module or exit the module with probability $q_{i\curvearrowright}$. Let p_α be the frequency with which node α is visited. If we then consider the possible states for a random walker within module i, the movement entropy within the module is given by

$$\begin{aligned} H(\mathcal{P}^i) = & - \frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta} \log_2\left(\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta}\right) && (2) \\ & - \sum_{\alpha\in i} \frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta} \log_2\left(\frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta}\right) && (3) \end{aligned}$$

Each entropy term $H(\mathcal{P}^i)$ is then weighted by the frequency of being in one of these states,

$$p^i_{\circlearrowright} = q_{i\curvearrowright} + \sum_{\alpha\in i} p_\alpha.$$

The full Map Equation (ME) is given by

$$L(M) = q_{\curvearrowright}H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i). \tag{4}$$

By minimizing this equation, the network description length is compressed while communities of nodes with higher information flow are identified.

3. ADDING NODE ATTRIBUTES

The Map Equation uses only information in links to partition the network into modules. However, networks are often *content-rich*, meaning that nodes have attributes associated with them. These attributes can provide more insight into the correct module classification of nodes, and they contribute to the description of information flow on a network. Take, for example, the world wide web. In addition to structure, given by hyperlinks between web pages, each page contains content, e.g., words, that differentiate it from other pages. Taking content into account gives a more robust view of the structure of the world wide web. Here we propose the Content Map Equation, which incorporates information about node attributes into the description of the random walk. This description can then be minimized to find modules in rich networks.

3.1 The Content Map Equation

We explicitly add the description length of node attributes into the Map Equation. We first consider a dictionary for node α, $\{d_j{}^\alpha\}$, that consists of attributes associated with the node. We then create a dictionary vector $\mathbf{x}^\alpha$ that gives the relative weight of each attribute for node α, i.e., $\sum_j x_j^\alpha = 1$. For example, when attributes are words from text associated with the node, the weight could simply be the frequency of each word. Next, we define a vector for each module, consisting of the dictionary vectors weighted by the node visit frequency, namely

$$x_j^{(i)} = \sum_{\alpha\in i} p_\alpha \; x_j^\alpha. \tag{5}$$

We examine the possible content states for the random walker within a module. The importance of attribute j in module i is given by $\frac{x_j^{(i)}}{p^{(i)}}$, where $p^{(i)} = \sum_{\alpha\in i} p_\alpha$. Thus, the average codeword length for the dictionary attributes within module i is bounded below by the entropy,

$$H(\mathcal{X}^i) \;\; = \;\; -\sum_j \frac{x_j^{(i)}}{p^{(i)}} \log_2\left(\frac{x_j^{(i)}}{p^{(i)}}\right). \tag{6}$$

This quantity is then weighted by the frequency of being in module i, $p^{(i)}$.

We add the term above to Eq. 4, resulting in the Content

Map Equation (CME):

$$L_C(M) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^{i}_{\circlearrowright} H(\mathcal{P}^i) + \sum_{i=1}^{m} p^{(i)} H(\mathcal{X}^i). \quad (7)$$

This gives the average description length of a step of an infinite random walk on a network with node attributes.

Note that this method has several desirable properties. The foremost advantage of the approach is its simplicity. It does only one thing — minimize the description length of a random walk — to partition the network using information from both links and node attributes. Furthermore, results do not depend on the number of attributes used to characterize nodes. This means that although a bad choice of representation (e.g., duplicating each attribute) would change the average description length, it will not affect partitioning results. Finally, since both links and attributes contribute equally to representing information in a network, our method does not require an additional parameter to control the contribution of each, in contrast to other methods [20, 24, 32, 35, 37].

3.2 Illustrative Examples

We demonstrate how adding content to the Map Equation can improve module division of a network with two illustrative examples. The first example consists of a clique, with one clique node connected to a chain of nodes, as shown in Figure 1(a). Dashed lines represent possible partitions of the network. Cut A bisects the network into two modules, grouping node 7 with the rest of its clique, whereas cut B groups it with the chain of nodes $1-6$. For simplicity, we use symbols to represent distinct nodes with d attributes described by vectors:

$$\bigcirc = (\underbrace{2/d,\cdots}_{d/2},\underbrace{0,\cdots}_{d/2}) \qquad \square = (\underbrace{0,\cdots}_{d/2},\underbrace{2/d,\cdots}_{d/2}) \quad (8)$$

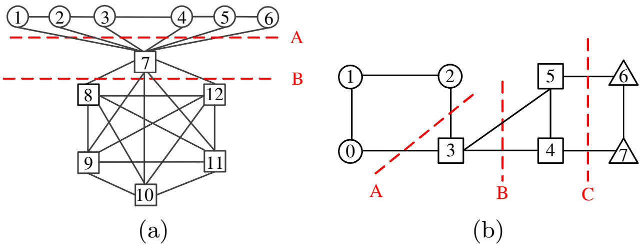


Figure 1: Example networks containing (a) two and (b) three different node types. Nodes of one type are described by a unique attribute vector. Dashed lines indicate possible partitions of the network.

Table 2 gives the number of bits required to describe different partitions of the network using the Map Equation and Content Map Equation for different number of node attributes. Let us first consider the case with $d = 4$ attributes and no cuts. It takes 3.41 bits to describe information contained in the links only using the Map Equation. On the other hand, it takes 1.89 bits to describe attributes alone, which indicates there is less information in the attributes than in the links.

Incorporating attributes changes the optimal partition of the network. Without attributes, the Map Equation prefers a balanced cut and chooses cut B over cut A or no cut at all, since it requires fewer bits (3.36). However, when attributes are incorporated into the Content Map Equation, cut A has a lower description length (4.59 bits) than cut B (4.87 bits). This partition is more consistent with our intuition for grouping node 7 with similar nodes in the clique.

Table 2: Minimum average description lengths of different partitions of the network in Figure 1(a) using links along (ME), attributes alone, or both links and attributes (CME) in the description of information flow. The number of node attributes is $d = 4$ or $d = 1,000$.

Cut	*links*	*attributes*		*both (CME)*	
	(ME)	$d = 4$	$d = 1000$	$d = 4$	$d = 1000$
no cut	3.41	1.89	10.86	5.30	14.27
A	3.59	**1.00**	**9.97**	**4.59**	**13.55**
B	**3.36**	1.51	10.47	4.87	13.83

Table 3: Minimum average description lengths of different partitions of the network in Fig. 1(b), using links along (ME), attributes alone, or both links and attributes (CME) in the description of information flow. The number of node attributes is $d = 4$ or $d = 1,000$.

Cut	*links*	*attributes*		*both (CME)*	
	(ME)	$d = 4$	$d = 1000$	$d = 4$	$d = 1000$
no cut	2.95	1.84	9.81	4.79	12.75
A	3.02	0.96	8.92	**3.98**	**11.95**
B	**2.93**	1.43	9.39	4.35	12.32
C	3.15	1.56	9.53	4.72	12.68
A+C	3.27	**0.80**	**8.77**	4.07	12.03
A+B	3.18	0.94	8.91	4.12	12.08
B+C	3.21	1.29	9.25	4.50	12.46
A+B+C	3.61	**0.80**	**8.77**	4.41	12.38

The Content Map Equation works correctly as the number of attributes grows. When there are $d = 1000$ attributes, it takes 10.86 bits to encode content alone, compared to 3.41 bits to describe links. However, it still prefers cut A over cut B.

Next we consider a more complex network in Figure 1(b) with three distinct node types, given by vectors:

$$\begin{aligned} \bigcirc &= (\underbrace{2/d,\cdots}_{d/2},\underbrace{0,\cdots}_{d/2}) \\ \square &= (\underbrace{0,\cdots}_{d/2},\underbrace{2/d,\cdots}_{d/2}) \\ \triangle &= (\underbrace{0,\cdots}_{3d/4},\underbrace{4/d,\cdots}_{d/4}) \end{aligned} \quad (9)$$

Note that circles and squares do not share any attributes, while squares and triangles share some attributes.

Table 3 gives the minimum average number of bits required to describe information in this network when taking into account only links, only content, and both links and content. Without the attributes, the Map Equation (ME) chooses cut B over the other options, partitioning this network into two modules. This seems like the natural, balanced division of the network. However, including content

Algorithm 1 Bottom-up search for Content Map Equation

```
Input: Network $G$
Output: A partition $M$ of $G$
 1: for each node $\alpha$
 2:     compute $p_\alpha$ and $\mathbf{x}^\alpha$
 3: Initialize $M$ by assigning each node to its own module
 4: $\Delta L = 0$
 5: do
 6:     let $M_{ij}$ denote a new partition resulting from merging
        modules $i$ and $j$ from $M$
 7:     $\{x,y\}$=$\arg\min_{i,\,j}$ $(L_C(M_{ij})$-$L_C(M))$
 8:     $\Delta L$=$L_C(M_{xy})$-$L_C(M)$
 9:     set $M$ as $M_{xy}$ if $\Delta L$ <0
10: while $\Delta L < 0$
11: return $M$
```

information changes the preferred partition of the network. While there are three distinct vectors, the nodes 3-7 share some of the attributes. Thus, when minimizing the Content Map Equation (CME) we find that the best solution is cut A with 3.98 bits when there are $d = 4$ attributes. This is different from either the cut preferred by links alone or the cut preferred by attributes alone. When nodes have many ($d = 1000$) attributes, cut A with 11.95 bits is still the preferred cut. Thus, partitioning results do not depend on the number of attributes used to characterize nodes.

4. FINDING MODULES

Minimizing the Content Map Equation is an NP-hard optimization problem. Similar to minimizing the matrix storage cost [1], the difficulty can be established by reducing it to the traveling salesman problem. To this end, we study feasible solutions from the realm of iterative heuristic algorithms. Rosvall & Bergstrom used an agglomerative (bottom-up) method that begins with each node in its own module and proceeds by greedily merging modules so as to decrease the description length. Unfortunately, even this greedy method is too computationally complex for larger networks. To address this issues, we further propose a scalable solution, namely top-down greedy search (see Section 4.2).

4.1 Bottom-up Method

We first consider a greedy agglomerative, or bottom-up, search algorithm [23], where each node is initially placed in its own module. Then, at each iteration, we merge two modules that result in the largest decrease in the Content Map Equation. This is repeated until there is no further benefit to merging modules. The details are presented in Algorithm 1.

In lines 1–2, we first calculate p_α and $\mathbf{x}^\alpha$ for each node. These quantities are constant and independent of the partition. The vector $\mathbf{x}^\alpha$ is chosen to give the weight of each attribute (or frequency of a word) associated with node α. If common attributes are shared by many nodes, then it may be more appropriate to use tf-idf weighing, lessening the importance of attributes associated with multiple classes.

The steady state of the node visit frequency of the infinite random walk, p_α, can be easily approximated for directed networks with the PageRank algorithm [3]. A small probability of teleportation to random nodes can be introduced to guarantee a unique steady state. Rosvall & Bergstrom [23] chose $\tau = 0.15$, which is equivalent to a damping factor of 0.85. For undirected networks, this node visit frequency is the relative sum of the edge weights incident to node α, compared to twice the full edge weight of the network, namely

$$p_\alpha = \frac{\sum_{\beta=1}^{n} A_{\alpha,\beta}}{\sum_{\beta=1}^{n}\sum_{\gamma=1}^{n} A_{\beta,\gamma}}, \qquad (10)$$

where A is the weighted adjacency matrix of the undirected network, with values corresponding to the edge weights between incident nodes.

After initialization, we start the greedy search (lines 5–10). The critical part is to compute the $L_C(M)$ (Eq. 7) for each possible partition M_{ij}, especially the exit probabilities for a given step $q_{i\curvearrowright}$, which can be easily calculated by

$$q_{i\curvearrowright} = \tau\left(\frac{n-n_i}{n-1}\right)\sum_{\alpha\in i} p_\alpha + (1-\tau)\sum_{\alpha\in i}\sum_{\beta\notin i} p_\alpha A_{\alpha,\beta} \qquad (11)$$

for directed networks and

$$q_{i\curvearrowright} = \sum_{\alpha\in i}\sum_{\beta\notin i} p_\alpha A_{\alpha,\beta} \qquad (12)$$

for undirected networks. Here, we take A to have row sums of one.

While this method does not provide the optimal solution to the minimization problem, it gets a reasonable approximation that identifies clusters of nodes with similar attributes as well as local structures.

4.1.1 Convergence analysis

We now briefly analyze the convergence property of the bottom-up algorithm. The Content Map Equation has both lower and upper bounds. In addition, the total cost of Eq. 7 is monotonically decreasing using Algorithm 1, since two modules are merged if and only if the total cost can be reduced, and the stopping criterion is satisfied if and only if the total cost cannot be reduced any further. Thus, the bottom up algorithm converges to a local optimum.

4.1.2 Complexity analysis

The computational complexity of each iteration of the bottom-up algorithm is $O(m^2(n + l + d))$, where m is the number of modules, n is number of nodes, l is number of links, and d is number of attributes. Hence, the total complexity of the partitioning procedure is $O(km^2(n + l + d))$, where k is number of iterations, which is usually a small number. Note that in the bottom-up algorithm, we start from the state where each node is a module, that is, in the worse case, $m = O(n)$.

4.2 Top-down Method

In the bottom-up method, we compute a better partition M with m modules from a partition M' with $m+1$ modules. However, for the initial state $m = n$, and the search space is essentially quadratic in the network size. For networks with a large number of nodes, the computational costs of even the greedy algorithm may be prohibitive. To address this problem, we propose a “top-down” search algorithm.

At first glance, it may be preferable to start with all nodes in the same module and proceed by splitting modules until no further decrease in the description length is achieved. However, in reality, this method can easily get trapped in local minima that do not represent a good partition of the

Algorithm 2 Top-down search for Content Map Equation

```
Input: Network $G$
Output: A partition $M$ of $G$
 1: Initialize $\Delta L = 0$, $M$, $p_\alpha$ and $\mathbf{x}^\alpha$
 2: for $i = 1$ to $\sqrt{n}$
 3:     randomly initialize partition $M'$ with $\sqrt{n}$ modules
 4:     set $M$ as $M'$ if reducing description length
 5: do
 6:     for each $\alpha$ in ordered node list $V$
 7:         let $M(\alpha)_i$ denote the new partition resulting
            from moving node $\alpha$ to an existing or a new
            empty module $i$
 8:         $x$=arg $\min_i$ $(L_C(M(\alpha)_i)$-$L_C(M))$
 9:         $\Delta L = L_C(M(\alpha)_x) - L_C(M)$
10:         set $M$ as $M(\alpha)_x$ if $\Delta L < 0$
11: while $\Delta L < 0$
12: return $M$
```

network. Instead, we start from a random configuration, with nodes assigned randomly to $m = \sqrt{n}$ modules. Note that this random configuration does not mean the number of modules found by the algorithm is $\sqrt{n}$ since both splitting (when a node is moved to a new empty module) and merging (all of the nodes in a module are assigned to another module) are considered in the algorithm. In each iteration of the search algorithm, a node is either assigned to a different existing module or a new empty module, whichever leads to a larger decrease in CME. The algorithm stops when it can no longer decrease the description length.

The top-down search is detailed in Algorithm 2. In lines 2–5, we create a random partition and choose the one with the smallest description length as the start state. While this heuristic is simple and naïve, it achieves better performance in real data than using LDA [2] or ME as initializations (see Section 5.3.3). Next, for each node α, we enumerate all possible improvements for α: assign α to either another existing module or a new empty module (lines 7–11). Another heuristic strategy in our algorithm is that we notice the previous correction for a node α might have influence for the latter correction of another node β. Hence, in each iteration, we order the node lists based on the descending order of p_α (line 8). The high-level intuition is that we want to find the improvements for those highly influential nodes first and then turn our attention to less influential nodes. The top-down search algorithm is guaranteed to converge to a local optimum. Convergence properties are similar to the bottom-up algorithm.

4.2.1 Complexity analysis

In each iteration, for each node α and each module i, we need to compute the change in description length, $L_C(M(\alpha)_i)$ -$L_C(M)$. This computation is very efficient, since only the source and target module is affected, thus the time complexity is $O(\delta_n + \delta_l + \delta_d)$. Here δ_n, δ_l, and δ_d denote the average number of nodes, links, and attributes respectively in the neighborhood of a node. Then the overall time complexity of our algorithm is $O(knm \times (\delta_n + \delta_l + \delta_d))$, where k is the number of iterations. In practice, however, both m and k are very small. Thus, our partitioning algorithm is efficient in most cases, as also verified by our experiments.

5. EVALUATION ON REAL-WORLD DATA

We use the Content Map Equation to partition real-world networks with known ground truth community labels. All of these networks are examples of content-rich networks in which nodes have attributes, such as content words for scientific papers in citation networks, or demographic features for Facebook users.

5.1 Data Sets

Twitter: We consider a network created by interactions among Twitter users on the subject related to proposition 30 on the November 2012 California ballot [27]. We used the method described in [27] to classify the position on the proposition of each user as for, against or neutral. These serve as the ground truth labels for these data.

For the attributes associated with each node (user), we considered the 25 hashtags used most frequently by that user and used tf-idf scores instead of hashtag frequency in the nodes' attribute vectors.

Facebook: We used a subset of a large social network of Facebook users that contains anonymized information about individuals, including hometown, gender, major, work, and year in school [16]. We took these features as attributes of each node. An edge represents a friendship between two users. For the ground truth community labels, we used the circles that have been identified in these data [16], with some users being members of multiple circles and other users not in any circle.

ArnetMiner: The ArnetMiner dataset is a citation network [29], classified according to research fields: data mining and association rules (DM), database systems and XML data (DB), information retrieval (IR), web services (WS), bayesian networks and belief function (BN), web mining and information fusion (WM), semantic web and description logics (SW), machine learning (ML), pattern recognition and image analysis (PR), natural language system and statistical machine translation (NLP). We used these class labels as ground truth data in the experiment. We treated words in the paper title as node attributes.

Citeseer: The CiteSeer dataset [25] is a citation network with 3312 scientific papers, classified into one of six classes, and 4732 links. Each paper is described by a 0/1-valued vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 3703 unique words.

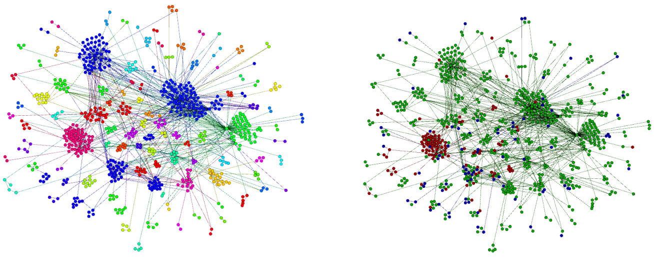
Pubmed: The Pubmed Diabetes dataset [25] contains 19,717 scientific publications from the PubMed database pertaining to diabetes, classified into one of three classes. Each publication in the dataset is described by a tf-idf weighted word vector from a dictionary which consists of 496 unique words.

Flickr: This dataset [17] was built by creating links between images from Flickr that share common metadata: images from the same location, submitted to the same gallery, group, or set, images taken by friends, etc. The attributes of a single node (image) include image features that are obtained from PASCAL [7], ImageCLEF [11], MIR [13], and NUS-wide[4]. We use the ground truth labels in the image classification tasks as the ground truth communities (only around 10% nodes have ground truth communities).

A set of selected statistics of all the above data are reported in Table 4.

Table 4: Statistics of datasets.

	#nodes	# links	# classes	# attributes
Twitter	565	1,008	3	24
ArnetMiner	2,555	6,101	10	4,214
Citeseer	3,312	4,536	6	3,703
Facebook	1,911	24,975	9	570
Pubmed	19,717	44,338	3	496
Flickr	105,938	2,316,948	215	26,041



(a) modules (b) ground truth

Figure 2: Largest component in the Proposition 30 user network. (a) Communities, indicated by different colors and locations, found by the Content Map Equation. (b) Ground truth labels of the users in the network in (a): for (green), against (red), neutral (blue)(best viewed in color.)

5.2 Qualitative Evaluation

We first look at the Twitter interaction network for Proposition 30. Figure 2(a) divides the network into communities according to the Content Map Equation, placing nodes within the same module closer together and with the same color. There are many small communities and a few larger, densely connected communities. But how far are the outputted communities from the ground truth?

The ground truth for this network is shown in Figure 2(b), where we placed nodes in the same locations, but colored them according to their stance: green for users who support Proposition 30, red for users who oppose it, and blue for neutral users. This highlights the types of communities found in this network, including a few large communities that predominantly consist of users of one stance and a few smaller communities comprised of individuals with difference stances. While the CME breaks users into many communities, the communities themselves are relatively pure, i.e., composed of users who have the same stance on the proposition.

Next, we visualize the communities found by CME in the ArnetMiner citations network. From Figure 3, we are able to observe that the communities are of two types, either a community with nodes of only one topic or a community with a mixture of topics. For instance, the majority of orange color nodes corresponds to the topic web mining and information fusion (WM). In communities of the latter kind, for instance, black dashed box with red color, it contains not only single-topic nodes of both (WS) and (DB), but also nodes that were members of both topics. One of the reason is that in the ground truth, some of the topics co-occur very frequently (i.e., there exists a single node with multiple ground truth topic labels). Thus, by qualitative evaluation, we verify that the partitioning outputted by CME correctly identifies similar nodes.

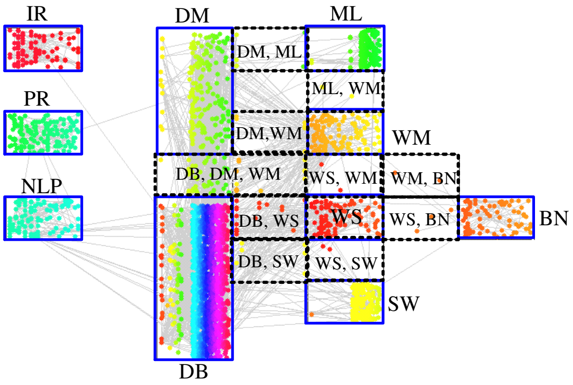


Figure 3: ArnetMiner citation network. Blue squares indicate different topics, with nodes inside the squares identified solely with that topic. Nodes inside dashed squares are associated with multiple topics and are placed between both topics with a hyphenated label. Color and location differentiate nodes in different communities found by the CME (best viewed in color).

5.3 Quantitative Evaluation

We quantitatively evaluate network partitioning using the point-wise normalized F-measure, purity and clustering accuracy to compare how well the discovered communities reproduce the classes present in the data.

F-measure.

Given an output community p and with reference to a ground truth class g (both in the form of node set), we define the precision rate as $|p \cap g|/|p|$ and the recall rate as $|p \cap g|/|g|$. The F-measure of p on g, denoted as $F(p, g)$, is the harmonic mean of precision and recall rates. The final F-measure [24] of the outputted partitioning P on the ground truth clustering G is then calculated as

$$F(P, G) = \sum_{p \in P} \left\{ \frac{|p|}{n} \max_{g \in G} F(p, g) \right\}. \qquad (13)$$

Purity.

The purity of the outputted partitioning P on the ground truth clustering G is defined as

$$\texttt{purity}(P, G) = \texttt{avg}_{p \in P} \left\{ \max_{g \in G} \frac{|p \cap g|}{|p|} \right\}. \qquad (14)$$

Clustering accuracy.

Assume that we assign the outputted community with ground truth label using the majority vote. Then the clustering accuracy evaluates the percentage of nodes with correct assignments.

$$A(P, G) = \frac{1}{n} \sum_{p \in P} \max_{g \in G} |p \cap g| \qquad (15)$$

We do not consider Normalized Mutual Information as a performance measure, because our approach finds a much larger number of classes than exist in the ground truth, making discovered classes poor predictors of the ground truth class distribution. If there are two groups with the same ground truth label but with no edges between them, then we shouldn't expect them to be placed in the same module.

Baselines.
We compare the algorithms proposed in this paper, bottom-up Content Map Equation (B-CME) and top-down CME (T-CME) to three classes of baselines: 1) content-based approaches, such as topic modeling (e.g., LDA [2]); 2) structure-based approaches such as the Map Equation (ME [23]); and 3) methods which use both links and attributes, such as BACG [32] and Codicil [24]. We do not compare to approaches [33, 36] since they only support a single-attribute per node. Our method produces much better F-measure scores than [37] on two benchmark dataset Citeseer and Pubmed. However, we do not include it in the results since they did not apply it to other datasets. The experiments were performed on a 2.7GHZ Intel i-7 CPU with 8G of memory.

5.3.1 Runtime

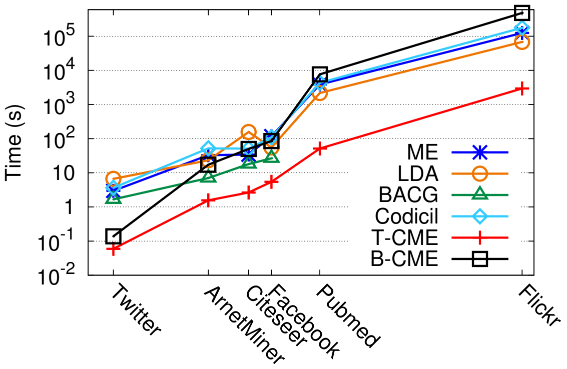


Figure 4: Runtime comparison of different algorithms. Data sets are ordered by their size (best viewed in color).

Figure 4 compares the runtimes of different methods. Results are ordered by network size (number of nodes and links, see Table 4), except for the Citeseer citations network, which we put between ArnetMiner and Facebook datasets to improve visualization.

Note that for baseline BACG, we only have the results for the small to medium-size networks, since the implementation of BACG runs out of memory for large networks such as Pubmed and Flickr. Results indicate that our bottom-up search implementation is faster than other baselines for small networks, comparable to other baselines for medium size network, but is much slower than other baselines for large networks. This motivates us to use the top-down implementation, which is significantly faster than alternative methods. The T-CME is about one order of magnitude faster than other baselines and two orders of magnitude faster than the B-CME.

For the baselines, BACG is more efficient than others since it stores many matrices in memory to facilitate computation, which leads to its memory bottleneck for large networks. The running time of content-based approach LDA depends on the number of nodes and the number of attributes in content vectors. Hence, LDA runs much faster than others in Flickr and Facebook, where the link information is much heavier than the content information. Codicil first constructs a content graph, performs local sparsity analysis on the content graph, and then runs the community detection algorithm ME on the sparse content graph. Thus, Codicil always runs slower than ME due to additional costs to construct and sparsify the content graph.

Table 5: Average Minimum Description length (MDL).

MDL	Twit ter	Arnet Miner	Cite seer	Face book	Pub med	Flickr
B-CME	7.780	12.354	12.033	13.524	15.870	15.79
T-CME	8.1436	12.596	11.601	14.120	15.809	16.092

5.3.2 Performance

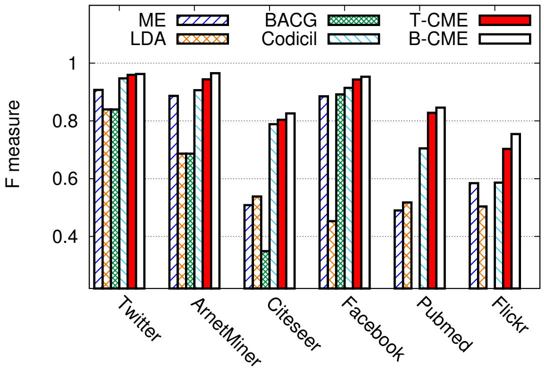


Figure 5: Performance comparison in terms of F measure (best viewed in color).

Figure 5 compares the F measures (Eq. 13) obtained by different approaches on the datasets. Recall that BACG fails to run the Pubmed and Flickr datasets on our machine due to huge memory consumption.

The results indicate that inclusion of node attributes lead to a better partition than using links alone (ME). The improvement is especially dramatic for the Citeseer and Pubmed datasets. The possible reason is that in the Citeseer and Pubmed citations networks, each node has very few links on average; therefore, structural information is very weak. Hence, the Map Equation finds a worse grouping of papers than content-aware approaches. The Content Map Equation is also much better than using content alone (e.g., LDA).

Compared to baselines BACG and Codicil, the Content Map Equation (both top-down and bottom-up) is consistently better. The top-down algorithm (T-CME) in general produces slightly worse results than the bottom-up approach, but it is much faster than the bottom-up method (B-CME). This indicates that the proposed top-down algorithm has a good trade-off between efficiency and quality.

In addition to F-measure, we also use purity (Eq. 14) and accuracy (Eq. 15), shown in Figure 6 and Figure 7, respectively, to evaluate network parittioning. Both results show that Content Map Equation outperforms the baseline BACG

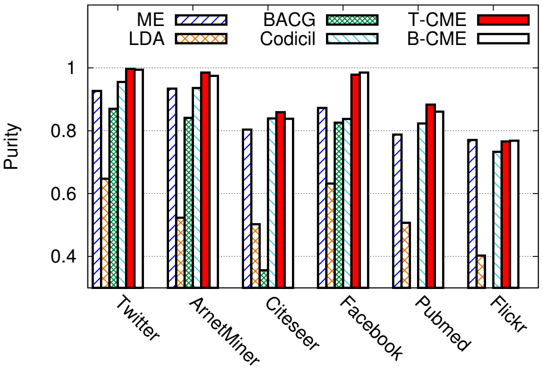


Figure 6: Performance comparison in terms of Purity (best viewed in color).

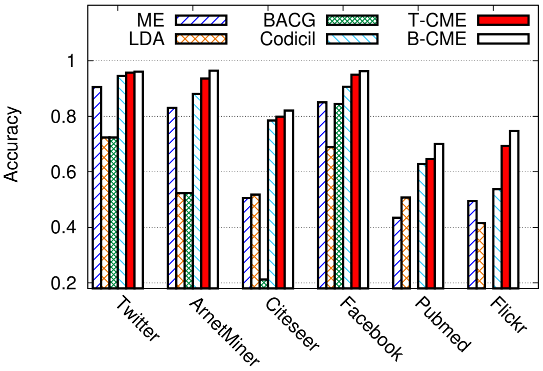


Figure 7: Performance comparison in terms of Accuracy (best viewed in color).

and Codicil. However, in Citeseer and Pubmed, the top-down greedy search (T-CME) produces a better partition than the bottom-up search (B-CME), according to the purity measure. There is no surprise if we look at the description length of the partitioning outputted by the two different search strategies (see Table 5). We notice that T-CME achieves lower description length than B-CME for Citeseer and Pubmed as well. These results are consistent with the intuition [31] that if we can correctly categorize the data (high purity within cluster), then the data can be described with the highest efficiency (i.e., using the minimum message length).

In summary, our approach identifies better communities in content-rich networks than alternative state-of-the-art methods that also take links and node attributes into account.

5.3.3 *Optimizations: Effect of Initialization*

Having established that the top-down method gives a good trade-off between partition quality and runtime, we now investigate the effect of different optimizations of the top-down algorithm. Specifically, we look at the effect of the initialization, i.e., the initial assignment of nodes to modules (see Section 4.2). We investigate whether leveraging attributes or links helps identify better modules. The intuition is that once the nodes are assigned to modules based on their attributes, CME can use information in the links to find a locally better solution. We use a topic modeling technique, e.g., LDA [2], to make the initial assignment. LDA requires the number of topics to be specified; hence, LDA20 means that the number of topics was set to 20, and LDA means that the number of topics was set to the true number of classes in the respective dataset. Alternatively, we can initialize the partition based on links alone, e.g., using the Map Equation, and then use attributes information to find a locally better solution with CME. We compare the partition quality resulting from random initialization to that resulting from LDA or ME initializations.

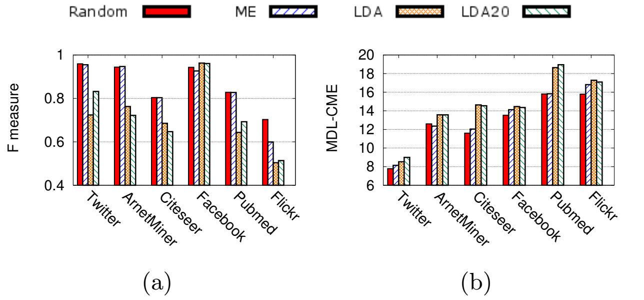


Figure 8: Comparison of the impact of different initializations on (a) F measure and (b) Minimum description length or the number of bits required by the Content Map Equation to describe the network partition.

Figure 8(a) reports the F-measure of the partition identified by the top-down method using different initializations (purity and accuracy results are similar). Surprisingly, the results demonstrate that neither LDA nor ME initializations help much in terms of partition quality improvement. Random initializations achieve better F-measure scores than LDA in 5 of 6 datasets, and better than ME initialization in 3 of 6 datasets. Since the Content Map Equation already incorporates content information equally with link information, the LDA/ME initializations only reweigh (or increase) the contribution of content/link information, which deteriorates performance.

Finally, we look at the effectiveness of different initialization methods to compress a random walk on a content-rich network. The results, shown in Figure 8(b), suggest that both LDA and ME initializations generally do not lead to better compression. Since both ME and LDA initialization are very time-consuming (see Figure 4), it is better to use random initialization in the top-down search method.

6. CONCLUSION

We have proposed and evaluated an information theoretic method for finding the modular structure of networks with node attributes. Building on the Map Equation of Rosvall & Bergstrom [23], we incorporate a new term that summarizes the contribution of the attributes to the description length of a random walk. By minimizing the resulting Content Map Equation, we are able to identify modules with a larger information flow among the nodes, where the nodes also have similar attributes.

Accounting for node attributes changes the discovered modules. Our empirical evaluation of several large real-world networks demonstrates that the Content Map Equation results in a partition that is closer to the ground truth division then using links alone, or using alternative methods that take attributes into consideration. Moreover, in contrast to other methods, our framework does not require adhoc parameters that control the contribution of links and attributes to structure. One drawback of the approach is that it does not capture the dependencies among attributes in module dictionaries. Because partitioning results are insensitive to, e.g., duplication of attributes in a representation, any additional information supplied by highly correlated attributes is essentially ignored. It would be an interesting challenge to extend the information theoretic framework to take these dependencies into account.

Acknowledgments

7. REFERENCES

[1] L. Akoglu, H. Tong, B. Meeder, and C. Faloutsos. Pics: Parameter-free identification of cohesive subgroups in large attributed graphs. In *SDM*, pages 439–450. SIAM, 2012.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30:107–117, 1998.

[4] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *CIVR*, 2009.

[5] F. R. K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, Feb. 1996.

[6] J. D. Cruz, C. Bothorel, and F. Poulet. Entropy based community detection in augmented social networks. In *CASoN*, pages 163–168. IEEE, 2011.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results.

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

[9] W. Fu, L. Song, and E. P. Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 329–336. ACM, 2009.

[10] K. Henderson, T. Eliassi-Rad, S. Papadimitriou, and C. Faloutsos. HCDF: A hybrid community discovery framework. In *SDM*, pages 754–7–65, 2010.

[11] T. D. Henning Müller, Paul Clough and B. C. (Eds.). *Experimental Evaluation in Visual Information Retrieval.* The Information Retrieval Series, Vol. 32, Springer, 2010.

[12] D. Huffman. A method for the construction of minimum-redundancy codes. *Proc. Inst. Radio Eng.*, 40(9):1098–1101, September 1952.

[13] M. J. Huiskes and M. S. Lew. The mir flickr retrieval evaluation. In *MIR*, 2008.

[14] J.-H. Kang and K. Lerman. Using lists to measure homophily on twitter. In *AAAI workshop on Intelligent Techniques for Web Personalization and Recommendation*, July 2012.

[15] B. Long, Z. M. Zhang, X. Wú, and P. S. Yu. Spectral clustering for multi-type relational data. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 585–592. ACM, 2006.

[16] J. McAuley and J. Leskovec. Learning to discover social circles in ego networks. *NIPS*, 2012.

[17] J. J. McAuley and J. Leskovec. In *ECCV (4)*, Lecture Notes in Computer Science, pages 828–841. Springer.

[18] M. Mcpherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444, 2001.

[19] M. E. J. Newman. Finding community structer in networks using the eigenvectors of matrices. *Physical Review E*, 74(3), 2006.

[20] G.-J. Qi, C. C. Aggarwal, and T. S. Huang. Community detection with edge content in social media networks. In *ICDE*, pages 534–545, 2012.

[21] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabási. Hierarchical Organization of Modularity in Metabolic Networks. *Science*, 297(5586):1551–1555, 2002.

[22] A. W. Rives and T. Galitski. Modular organization of cellular networks. *Proc Natl Acad Sci U S A*, 100(3):1128–1133, 2003.

[23] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, Jan. 2008.

[24] Y. Ruan, D. Fuhry, and S. Parthasarathy. Efficient community detection in large networks using content and links. In *WWW*, pages 1089–1098, 2013.

[25] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[26] C. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 1948.

[27] L. M. Smith, L. Zhu, K. Lerman, and Z. Kozareva. The role of social media in the discussion of controversial topics. In *ASE/IEEE International Conference on Social Computing*, 2013.

[28] D. A. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. *Linear Algebra and its Applications*, 421(2-3):284–305, Mar. 2007.

[29] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. *In Proceedings of the Fifteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009.

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

[31] C. S. Wallace and D. M. Boulton. An information measure for classification. *Computer Journal*, 11(2), 1968.

[32] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng. A model-based approach to attributed graph clustering. In *SIGMOD Conference*, pages 505–516, 2012.

[33] J. Yang, J. McAuley, and J. Leskovec. Community detection in networks with node attributes. In *International Conference On Data Mining (ICDM)*. IEEE, 2013.

[34] T. Yang, R. Jin, Y. Chi, and S. Zhu. Combining link and content for community detection: a discriminative approach. In *KDD*, pages 927–936, New York, NY, USA, 2009. ACM.

[35] Y. Zhou, H. Cheng, and J. X. Yu. Graph clustering based on structural/attribute similarities. *PVLDB*, 2(1):718–729, 2009.

[36] L. Zhu, W. K. Ng, and J. Cheng. Structure and attribute index for approximate graph matching in large graphs. *Inf. Syst.*, 36(6):958–972, 2011.

[37] Y. Zhu, X. Yan, L. Getoor, and C. Moore. Scalable text and link analysis with Mixed-Topic link models. In *Proc. of KDD*, 2013.