Partitioning Networks with Node Attributes by Compressing Information Flow

LAURA M. SMITH, California State University, Fullerton
LINHONG ZHU and KRISTINA LERMAN, University of Southern California
ALLON G. PERCUS, Claremont Graduate University

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 distinguished 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 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.

CCS Concepts: • **Information systems → Social networking sites**; **Clustering**; • **Human-centered computing → Social networks;**

Additional Key Words and Phrases: Graph partitioning, community detection, information theory, rich networks

ACM Reference Format:
Laura M. Smith, Linhong Zhu, Kristina Lerman, and Allon G. Percus. 2016. Partitioning networks with node attributes by compressing information flow. ACM Trans. Knowl. Discov. Data 11, 2, Article 15 (November 2016), 26 pages.
DOI: http://dx.doi.org/10.1145/2968451

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 [Ravasz et al. 2002; Rives and Galitski 2003] or communities in social networks [Newman 2006]. 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 [Chung 1996; von Luxburg 2007; Spielman and Teng 2007] identify which links to cut to separate the

The work was supported in part by AFOSR (contract FA9550-10-1-0569), by DARPA (contract W911NF-12-1-0034), and by the NSF (under grants CIF-1217605 and SMA-1360058).
Authors' addresses: L. M. Smith, Department of Mathematics, California State University, Fullerton, CA; email: lausmith@fullerton.edu; L. Zhu and K. Lerman, Information Sciences Institute, University of Southern California, Marina del Rey, CA; emails: linhong.seba.zhu@gmail.com, lerman@isi.edu; A. G. Percus, Claremont Graduate University, Claremont, CA; email: allon.percus@cgu.edu.

DOI: http://dx.doi.org/10.1145/2968451

network into disconnected components, while modularity-based approaches [Newman 2006; Fortunato 2010] find clusters of densely connected nodes. Real-world networks, however, are often rich in content, with nodes that have distinguished features or attributes. Individuals in a social network differ in age, gender, education, and interests, whereas 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 [Mcpherson et al. 2001], and taking them into account may improve the quality of the discovered modules. This observation has inspired several attempts to partition content-rich networks [Qi et al. 2012; Xu et al. 2012; Ruan et al. 2013; Yang et al. 2009; Zhou et al. 2009; Zhu et al. 2011, 2013]. 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.

We approach the problem of identifying the modular structure of networks using information theory [Rosvall and Bergstrom 2008]. This approach is based on an observation that information flow on a network tends to get trapped within modules. As an example, consider a random walk, a popular proxy for modeling the flow of information on a network. If clusters exist within a network, they tend to trap the random walk, slowing its mixing rate [Jerrum and Sinclair 1988]. This is the basis for spectral partitioning algorithms [Chung 1996; Smith et al. 2013a] that divide the network into disjoint clusters by cutting edges. From an information-theoretic view point, on the other hand, the existence of clusters or modules means that it is possible to compress the description of a random walk on a network by reusing names of nodes in different modules. The duality between compression and structure identification allows us to partition the network into modules by minimizing the expected description length of the flow of information, e.g., a random walk, on the network [Rosvall and Bergstrom 2008]. 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 also need an effective means of accounting for the attributes of nodes. To this end, we introduce the *Content Map Equation* (CME), which incorporates node attributes into a description of information flow, and use it to compress the flow of information on content-rich networks. The CME groups nodes into modules not only when information frequently flows between them, but also when they have similar attributes.

Finding a minimum solution to the CME is a hard optimization problem. Similar to Rosvall and Bergstrom, we use a greedy bottom-up search to find a locally optimal solution [Rosvall and Bergstrom 2008]. In this procedure, each node starts in its own module, and the search proceeds by merging modules so as to minimize the total description length. However, the approach 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 real-world networks with node attributes and a known modular structure. We demonstrate the CME identifies better modules than approaches that do not use content information. We also show 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 and Bergstrom's method. In Section 3, we introduce the CME 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 CME 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

Table I. Summary of Related Work on Community Detection Using Both Links and Node Attributes

	Method	Multi-attribute	Parameter-free	$O(km^2(n+l+d))$ and above	$O(km(l+nz))$ or $O(km(l+nd))$
Generative	Henderson et al. [2010]	✓	–	✓	–
	Xu et al. [2012]	–	–	?	–
	Yang et al. [2013]	✓	–	–	✓
	Yang et al. [2009]	–	–	✓	–
	Zhu et al. [2013]	✓	–	–	✓
Hybrid	Ruan et al. [2013]	✓	–	?	–
	Zhou et al. [2009]	–	–	✓	–
	Zhu et al. [2011]	✓	✓	–	✓
MF	Long et al. [2006]	✓	–	✓	–
	Qi et al. [2012]	✓	✓	?	–
IR	Akoglu et al. [2012]	✓	✓	✓	–
	Cruz et al. [2011]	✓	✓	?	–
Other	Ugander and Backstrom [2013]	✓	?	–	✓
	Günnemann et al. [2013]	✓	–	?	–
Proposed	B-CME	✓	✓	✓	–
	T-CME	✓	✓	–	✓

Notes: n: number of nodes, l: number of links, d: number of attributes, k: number of iterations, m: number of communities, and $nz <= nd$ is number of non-zero entries in node attribute matrix. MF and IR stand for matrix factorization and information theory-based approaches. The question mark "?" denotes "unknown or depends on parameters." The last two rows list features of proposed methods.

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 by using both links and node attributes. Proposed techniques range from generative modeling, to matrix factorization, to information theoretic approaches. See Table I for a (non-exhaustive) summary.

Most of the existing generative modeling approaches, such as Yang et al. [2009], Henderson et al. [2010], Xu et al. [2012], and Zhu et al. [2013], extend the mixed-membership model with the assumption that communities and attributes together generate links. In contrast, Yang et al. [2013] 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, whereas all remaining approaches using generative modeling in Table I support multi-dimensional attributes.

Another popular category for community detection methods using both links and content is the hybrid approach [Ruan et al. 2013; Zhou et al. 2009; Zhu et al. 2011; Silva et al. 2012; Moser et al. 2009]. The general workflow of the approach is as follows: it generates content links based on attribute vector similarity and then combines content links with topological links to detect communities. One of the more successful techniques used for this approach has been spectral clustering [Günnemann et al. 2013] which, like our work, is based on the behavior of random walks, though applies different clustering techniques to this behavior.

Compared to generative modeling and hybrid approaches, fewer methods have been developed that use matrix factorization or information theory. Matrix factorization [Long et al. 2006; Qi et al. 2012] aims to jointly co-factorize the adjacency matrix of the graph and the node–attribute matrix to obtain the low-ranked node-community matrix.

Our work is also related to graph partitioning algorithms that utilize attribute information. For instance, Ugander and Backstrom [2013] proposed a label propagation algorithm to partition users in a friendship network into different partitions while preserving edge locality. They also utilized the attribute information of users, such as geo-location, to improve the partitioning quality.

From the information theoretic view, the entropy-based approach [Cruz et al. 2011] aims to detect communities with low entropy and high modularity. Akoglu et al. [2012] extracted cohesive subgraphs by compressing the storage cost of matrices. The storage cost of matrices is related to the universal code length of integers (i.e., number of clusters/nodes/attributes) and the Shannon entropy of clustering mapping matrices. This is different from our encoding for random walks (see Section 2.1 for more details). We approach the problem of partitioning of content-rich networks from a different information theoretic perspective: exploiting the duality between identifying communities and compressing both content and structure information, which differs from the matrix storage compression [Akoglu et al. 2012]. Our method is inspired by Rosvall and Bergstrom [2008], who proposed compressing information flows on a network in order to identify modules. By minimizing the *ME*, their method identifies modules with large internal information flows. Below, we briefly describe compressing random walks using the ME.

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 [Huffman 1952]. 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 [Shannon 1948], 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 ME, 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 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 ME [Rosvall and Bergstrom 2008] gives the average description length for a step of an infinite random walk through 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 for both levels. The ME is expressed in terms of the entropy $H(\mathcal{Q})$ of the module-level codewords, specifying module changes, and the entropies $H(\mathcal{P}^i)$ of the node-level codewords for each module i, specifying movements from nodes within that module. Each of these entropy terms is weighted by the rate at which the corresponding codewords are used.

Given partition M of n nodes into m modules, let $q_{i\curvearrowright}$ be the probability of exiting module i. Then,

$$q_{\curvearrowright} = \sum_{i=1}^{m} q_{i\curvearrowright}$$

specifies the rate at which module-level codewords are used. We next look within each module, at the node level, 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 (a step that needs its own codeword) with probability $q_{i\curvearrowright}$. Let p_α be the probability of visiting node α. Then,

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

specifies the rate at which node-level codewords are used in module i. Consequently, the ME is given by

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

Minimizing $L(M)$ compresses the network description length while identifying communities of nodes with higher information flow.

The entropies themselves are straightforward. Among module-level steps, the frequency of changing to module i is $q_{i\curvearrowright}/q_{\curvearrowright}$, so

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

Among node-level steps, the frequency of exiting module i is $q_{i\curvearrowright}/p^i_{\circlearrowright}$, and the frequency of visiting state α is $p_\alpha/p^i_{\circlearrowright}$, so

$$H(\mathcal{P}^i) = -\frac{q_{i\curvearrowright}}{p^i_{\circlearrowright}}\ \log_2\left(\frac{q_{i\curvearrowright}}{p^i_{\circlearrowright}}\right) - \sum_{\alpha\in i}\frac{p_\alpha}{p^i_{\circlearrowright}}\ \log_2\left(\frac{p_\alpha}{p^i_{\circlearrowright}}\right).$$

The steady state of the node visit frequency of the infinite random walk, p_α, can be approximated for directed networks with the PageRank algorithm [Brin and Page 1998]. A small probability of teleportation to random nodes can be introduced to guarantee a unique steady state. Rosvall and Bergstrom [2008] 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,

$$p_\alpha = \frac{\sum_{\beta=1}^{n} A_{\alpha,\beta}}{\sum_{\beta=1}^{n}\sum_{\gamma=1}^{n} A_{\beta,\gamma}}, \tag{2}$$

where A is the weighted adjacency matrix of the undirected network, with values corresponding to the edge weights between incident nodes. We take A to have row sums of one. The Personalized PageRank could also be used to determine p_α.

The exit probabilities for a given step, $q_{i\curvearrowright}$, can be calculated for directed networks by

$$q_{i\curvearrowright} = \tau\left(\frac{n-n_i}{n-1}\right)\sum_{\alpha\in i} p_\alpha + (1-\tau)\sum_{\alpha\in i}\sum_{\beta\notin i} p_\alpha A_{\alpha,\beta}, \tag{3}$$

and for undirected networks by

$$q_{i\curvearrowright} = \sum_{\alpha\in i}\sum_{\beta\notin i} p_\alpha A_{\alpha,\beta}. \tag{4}$$

We note that one could also formulate the model using probabilities of remaining in a given module, leading to a different way of describing the network. This would give a different description length formulation. By using exit probabilities, we specify the codeword for the random walker's module only when changing modules.

Finally, note the distinction between the ME approach and spectral clustering [von Luxburg 2007]. Both approaches aim at discovering modules (communities) by identifying regions of the network from which a random walk rarely escapes. However, the ME then identifies these regions through a two-level description that takes into account both visiting frequencies p_α and exit probabilities $q_{i\curvearrowright}$, with the number of resulting communities chosen so as to minimize description length. Spectral clustering identifies these regions based only on the visiting frequencies p_α and chooses a fixed number of communities, either through k-means or through a hierarchical approach.

2.3. Relationship to Conditional Entropy

There is a relationship between the second term of the ME and conditional entropy. Let $\mathcal{R}$ be the set of $(m+n)$ possible node-level events, for either exiting a module or visiting a node. Let $\mathcal{S}$ be the set of m possible module-level events, each of which consists of all the outcomes associated with exiting or remaining within that module (Note that this is distinct from the set $\mathcal{Q}$ of module-level events associated with module *changes*). Then, assigning frequencies to these events according to the frequencies of using the corresponding codewords, we can show that

$$\sum_{i=1}^{m} p_{\circlearrowright}^{i} H(\mathcal{P}^i) = (1 + q_{\curvearrowright}) H(\mathcal{R}|\mathcal{S}), \tag{5}$$

where $1 + q_{\curvearrowright} = \sum_i p_{\circlearrowright}^{i}$, and $H(\mathcal{R}|\mathcal{S})$ denotes the conditional entropy.

Thus, the contribution to the ME by the node-level entropy term can be interpreted as the entropy of a node-level description of the random walk conditioned on a module-level description, weighted by the rate at which node-level codewords are used. This interpretation will be useful in our subsequent discussion of content on networks.

3. ADDING NODE ATTRIBUTES

The ME uses only information in links to partition the network into modules. However, networks are often *content-rich*, meaning 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 the structure of hyperlinks between web pages, each page contains content, e.g., words, that differentiates it from others. Taking content into account gives a more robust view of the structure of the world wide web. Here, we propose the CME, which incorporates node attribute information 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 ME. We first consider a dictionary of length d for node α, consisting of d 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., $\sum_j x_j^\alpha = 1$. The quantity $\mathbf{x}^\alpha$ is 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: x_j^α may be interpreted as the rate at which a random walk encounters attribute j while visiting node α.

Next, we define a vector for each module, consisting of the dictionary vectors weighted by the node visit frequency, namely

$$x_j^{(i)} = \sum_{\alpha \in i} p_\alpha \, x_j^\alpha .$$

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

$$H(\mathcal{X}^i) = -\sum_{j=1}^{d} \frac{x_j^{(i)}}{p^{(i)}} \log_2 \left(\frac{x_j^{(i)}}{p^{(i)}} \right).$$

We note that if $\sum_j x_j^\alpha > 1$, the entropy $H(\mathcal{X}^i) < 0$. Therefore, we must have $\sum_j x_j^\alpha = 1$ to be able to find the lower bound of the average codeword length.

The quantity $H(\mathcal{X}^i)$ is then weighted by the rate at which codewords for attributes within module i are used,

$$\sum_{j=1}^{d} x_j^{(i)} = p^{(i)}.$$

Note that we do not account for module exits in this rate, as no attributes are associated with them.

We add the new weighted term to Equation (1), resulting in the CME:

$$L_C(M) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p_{\circlearrowright}^{i} H(\mathcal{P}^i) + \sum_{i=1}^{m} p^{(i)} H(\mathcal{X}^i). \tag{6}$$

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

The CME is a straightforward quantity in the description of the network with node attributes. The minimum of $L_C(M)$ would give the optimal compression of the network, revealing modular structure. The average description length of a random walk may not be the optimal quantity to minimize, but it works well in practice.

3.2. Relationship to Conditional Entropy

Similar to the relationship of the ME's second term to conditional entropy, there is an analogous interpretation for the third term of the CME. Let $\mathcal{X}$ be the set of d possible node-level events for encountering a given attribute. Let $\mathcal{T}$ be the set of m possible module-level events, each of which consists of all the outcomes associated with remaining within that module (though not exiting, since attributes are associated only with nodes within modules and not with module exits). Then,

$$\sum_{i=1}^{m} p^{(i)} H(\mathcal{X}^i) = H(\mathcal{X}|\mathcal{T}). \tag{7}$$

Therefore, the CME contribution for the attributes is the entropy of a node-level description of attributes encountered by the random walk, conditioned on a module-level description. Not surprisingly, the frequency of encountering an attribute is simply

$$\sum_{i=1}^{m} \sum_{j=1}^{d} x_j^{(i)} = \sum_{i=1}^{m} p^{(i)} = 1,$$

so the conditional entropy $H(\mathcal{X}|\mathcal{T})$ is effectively weighted by the rate at which attributes are encountered. Thus, the CME can be rewritten as

$$L_C(M) = q_{\curvearrowright} H(\mathcal{Q}) + (1 + q_{\curvearrowright}) H(\mathcal{R}|\mathcal{S}) + H(\mathcal{X}|\mathcal{T}),$$

where the coefficients of each term represent the rates of using codewords for the corresponding level of a three-level description: modules, nodes, and attributes.

Note that the notion of $\mathcal{T}$ differs from $\mathcal{S}$ defined in Section 2.3, in that the events of $\mathcal{T}$ are restricted to be within each module while the events of $\mathcal{S}$ contain both within-module movements and exit-module movement. In the equation above, the node-level description is conditioned on $\mathcal{S}$ because node-level movements allow for module exit, whereas the attribute-level description is conditioned on $\mathcal{T}$ because attribute-level movements are internal to a module.

3.3. Properties of the Content Map Equation

This method has several notable 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. In our formulation of the problem, both links and attributes contribute equally to representing information in a network, in contrast to other methods [Qi et al. 2012; Ruan et al. 2013; Xu et al. 2012; Zhou et al. 2009; Zhu et al. 2013]. If needed, our method

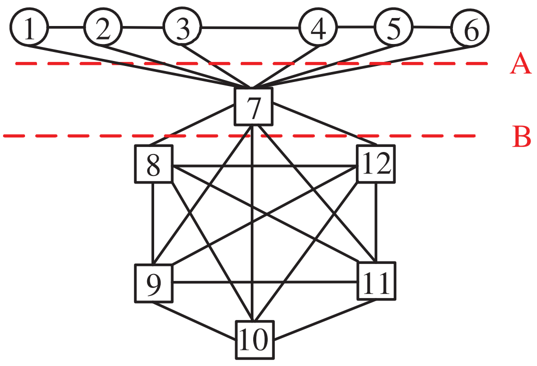


Fig. 1. Example networks containing two different node types. Nodes of one type are described by a unique attribute vector. Dashed lines indicate possible partitions of the network.

allows for adjusting the relative importance of links and attributes by using an additional parameter to control the contribution of each. Furthermore, results are in some cases insensitive to the exact choice of attribute representation used to characterize nodes. While a bad choice of representation may affect the average description length, some other choices, such as duplicating attributes, will not change partitioning results. To see why, consider a network with K attributes per node. We increase the number of attributes by a factor of γ to $d = \gamma K$ attributes. Here, we take an attribute with weight w_j and repeat it γ times with a new weight of $\frac{w_j}{\gamma}$, distributing the weight of one attribute equally over γ attributes by the mapping

$$(w_1, \dots, w_K) \mapsto \Bigg(\underbrace{\frac{w_1}{\gamma}, \dots, \frac{w_1}{\gamma}}_{\gamma \text{ times}}, \dots, \underbrace{\frac{w_K}{\gamma}, \dots, \frac{w_K}{\gamma}}_{\gamma \text{ times}} \Bigg).$$

Proposition. *Let $L_C(M)$ and $L_C^\gamma(M)$ denote the CME evaluated on a partition M with respective attribute vectors of the form*

$$\mathbf{w} = (w_1, \dots, w_K) \quad \textit{and} \tag{8}$$

$$\mathbf{w}_\gamma = \left(\frac{w_1}{\gamma}, \dots, \frac{w_1}{\gamma}, \dots, \frac{w_K}{\gamma}, \dots, \frac{w_K}{\gamma} \right). \tag{9}$$

Then, for $\Delta L = L_C(M_1) - L_C(M_2)$ *and* $\Delta L^\gamma = L_C^\gamma(M_1) - L_C^\gamma(M_2)$, $\quad \Delta L = \Delta L^\gamma$.

Proof. The proof is presented in Appendix A.1. □

This proposition shows that the relative difference between two partitions is identical for the attribute vectors in Equations (8) and (9), indicating that by duplicating or multiplying the number of attributes uniformly in this fashion, the optimal partitioning does not change when minimizing the CME. Of course, if only certain attributes are duplicated, this could introduce a bias into the representation that may affect partitioning results.

3.4. An Illustrative Example

We demonstrate how adding content to the ME 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. Dashed lines represent

Table II. Minimum Average Description Lengths of Different Partitions of the Network in Figure 1 Using Links Along (ME), Attributes Alone, or Both Links and Attributes (CME) in the Description of Information Flow

Cut	Links (ME)	Attributes $d = 4$	Attributes $d = 1{,}000$	Both (CME) $d = 4$	Both (CME) $d = 1{,}000$
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

Notes: The number of node attributes is $d = 4$ or $d = 1{,}000$.

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:

$$\bigcirc = (\underbrace{2/d, \ldots, 2/d}_{d/2}, \underbrace{0, \ldots, 0}_{d/2}), \qquad \square = (\underbrace{0, \ldots, 0}_{d/2}, \underbrace{2/d, \ldots, 2/d}_{d/2}). \tag{10}$$

Table II gives the number of bits required to describe different partitions of the network using the ME and CME 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 ME. 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 ME 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 CME, 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.

The CME works correctly as the number of attributes grows. When there are $d =$ 1,000 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. In addition, given any two different cuts P_1 and P_2, $\Delta L(P_1^{d=4}, P_2^{d=4}) = \Delta L(P_1^{d=1{,}000}, P_2^{d=1{,}000})$ (possible differences of 0.01 in ΔL for Table II are due to rounding). Thus, partitioning results do not change when the number of attributes used to characterize nodes is scaled uniformly in this way.

Next, we consider a more complex network in Figure 2 with three distinct node types, given by vectors

$$\begin{aligned} \bigcirc &= (\underbrace{2/d, \ldots, 2/d}_{d/2}, \underbrace{0, \ldots, 0}_{d/2}), \\ \square &= (\underbrace{0, \ldots, 0}_{d/2}, \underbrace{2/d, \ldots, 2/d}_{d/2}), \\ \triangle &= (\underbrace{0, \ldots, 0}_{3d/4}, \underbrace{4/d, \ldots, 4/d}_{d/4}). \end{aligned} \tag{11}$$

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

Table III 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 ME chooses cut B over the other options,

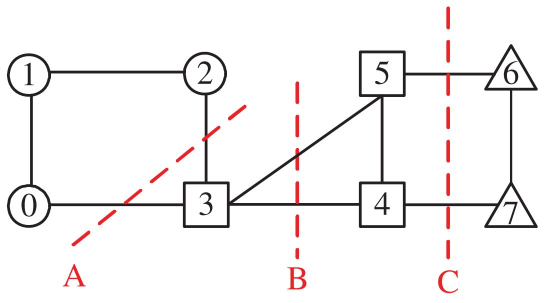


Fig. 2. Example networks containing three different node types. Nodes of one type are described by a unique attribute vector. Dashed lines indicate possible partitions of the network.

Table III. Minimum Average Description Lengths of Different Partitions of the Network in Figure 2, Using Links Along (ME), Attributes Alone, or Both Links and Attributes (CME) in the Description of Information Flow

Cut	Links (ME)	Attributes		Both (CME)	
		$d = 4$	$d = 1{,}000$	$d = 4$	$d = 1{,}000$
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

Notes: The number of node attributes is $d = 4$ or $d = 1{,}000$.

partitioning this network into two modules. This seems like the natural, balanced division of the network. However, including content 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 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 ($d = 1{,}000$) attributes, cut A with 11.95 bits is still the preferred cut. Thus, we once again see that partitioning results do not change when the number of attributes used to characterize nodes is scaled uniformly.

4. FINDING MODULES

Minimizing the CME is an NP-hard optimization problem. Similar to minimizing the matrix storage cost [Akoglu et al. 2012], the complexity can be established by a reduction from the traveling salesman problem. Consequently, we study feasible solutions from the realm of iterative heuristic algorithms. Rosvall and 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 issue, we further propose a scalable solution, namely top-down greedy search (see Section 4.2). Like the bottom-up approach, it provides at best a locally optimal solution.

ALGORITHM 1: Content Map Equation Bottom-up Search

```
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 $M$
```

4.1. Bottom-up Method

We first consider a greedy agglomerative, or bottom-up, search algorithm [Rosvall and Bergstrom 2008], in which each node is initially placed in its own module. Then, at each iteration, we merge the two modules that result in the largest decrease in the CME. This is repeated until there is no further benefit to merging modules. See Algorithm 1 for details.

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.

Convergence analysis. We now briefly analyze convergence of the bottom-up algorithm. The CME has both lower and upper bounds. In addition, the total cost of Equation (6) 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.

Complexity analysis. The computational complexity of each iteration of the bottom-up algorithm is $O(m^2(n + l + d))$, where m, n, l, and d are the numbers of modules, nodes, links, and attributes, respectively, as defined in Table I. The total complexity of partitioning is $O(km^2(n + l + d))$, where k is the number of iterations, which is usually a small number. Note that in the bottom-up algorithm, we start from a state in which each node is in its own module, which is $m = O(n)$ in the worst case.

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 network. Instead, we start from a relative better random configuration, with nodes assigned randomly to $m = \sqrt{n}$ modules. Inspired by Choi et al. [2012], we choose $m = \sqrt{n}$ since they have shown that the number of communities is allowed to grow no faster than $\sqrt{n}$. In addition, 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.

ALGORITHM 2: Content Map Equation Top-down Search

```
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 $M$
```

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. Although this heuristic is simple and naïve, it achieves better performance in real data than using LDA [Blei et al. 2003] or ME as initializations (see Section 6). Next, for each node α, we enumerate all possible improvements for α to find the best local movement for α: we assign α to either another existing module or a new empty module i if it leads to the greatest reduction in description length (lines 7–11). Another heuristic strategy is that the previous correction for a node α might have influence on a later correction of another node β. Hence, in each iteration, we order the node lists based on the descending order of p_α (line 6). The high-level intuition is that we want to find the improvements for those highly influential nodes first and then turn 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.

Note that in each iteration, for each node α and module i, we need to compute the change in description length, $L_C(M(\alpha)_i) - L_C(M)$ (Lines 6–9). This computation is very efficient since only the source and target module are affected. Specifically, for each node α, if it is moved from module i to module j, we first need to update $q_{i\curvearrowright}$ for module i and $q_{j\curvearrowright}$ for module j, which can be computed efficiently. For example, in undirected networks, the new $q_{i\curvearrowright}$ can be computed via the following equation:

$$q_{i\curvearrowright} = q_{i\curvearrowright}^{\text{old}} - \sum_{\beta \in N(\alpha), \beta \in j} A_{\alpha\beta} + \sum_{\beta \in N(\alpha), \beta \in i} A_{\alpha\beta}, \tag{12}$$

where $N(\alpha)$ denotes the set of neighbors of node α.

With the updated $q_{i\curvearrowright}$ and $q_{j\curvearrowright}$, all the terms used in describing link information such as $H(\mathcal{P}^i)$ and $H(\mathcal{Q})$ can be computed efficiently with the previous values for each term.

Table IV. Statistics of Datasets

	# nodes	# links	# classes	# attributes
Twitter	565	1,008	3	25
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

Similarly, the changes in description length of content, can be computed as follows:

$$\Delta_c = (p^{(i)} - p_\alpha)H(\mathcal{X}^i_{-\alpha}) + (p^{(j)} + p_\alpha)H(\mathcal{X}^j_{+\alpha}) - p^{(i)}H(\mathcal{X}^i) - p^{(j)}H(\mathcal{X}^j), \qquad (13)$$

where $H(\mathcal{X}^i_{-\alpha})$ is the average code length of attributes in module i after excluding node α, and $H(\mathcal{X}^j_{+\alpha})$ is the average code length of attributes in module j after including the new node α.

Complexity analysis. For each iteration, as shown in Equation (12), the time complexity to compute the changes in description length for each node α and module i is $O(|N(\alpha)|)$. In order to compute the changes in description length of content, the set of attributes of node α must be accessed (see Equation (13)). Thus, the computational cost is $O(|D(\alpha)|)$, where $D(\alpha)$ represents the set of non-zero attributes of α. Therefore, the time complexity of each iteration is $O(m\sum_\alpha(|N(\alpha)| + |D(\alpha)|))$. Since $\sum_\alpha |N(\alpha)|$ is equal to twice the number of edges l in graph, and $\sum_\alpha |D(\alpha)|$ is the number of all the non-zero entries in the node attribute matrix, which we call nz (and is much less than nd), the overall time complexity of our algorithm for k iterations is $O(km(l + nz))$ or $O(km(l + nd))$. In practice, 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 CME to partition real-world networks with known ground truth community labels. All of these networks are content-rich networks in which nodes have attributes, such as content words for scientific papers in citation networks or demographic features for Facebook users. Specifically, in the following experiments, we use a Twitter network [Smith et al. 2013b], ArnetMiner citation network [Tang et al. 2009], Citeseer citation network [Sen et al. 2008], Facebook friendship network [McAuley and Leskovec 2012], Pubmed network [Sen et al. 2008], and Flickr affinity network [McAuley and Leskovec 2012]. A set of selected statistics of all the data above are reported in Table IV, whereas a more detailed description of each dataset is presented as follows.

Note that content attributes may be represented with a normalized bag-of-words (BOW) representation, or alternatively with normalized weighted vectors. When attributes are words from text associated with the node, the weight can simply be the frequency of each word. When common attributes are shared by many nodes, we find it helpful to use tf-idf weighting, lessening the importance of attributes associated with multiple classes. For instance, using tf-idf weighting here could help eliminate common stop words. Theoretically, compressing the CME will already lead to shorter description lengths for stop words since they appear frequently across all modules, so in principle the minimum description length should not be sensitive to the use of tf-idf. But in practice, using tf-idf weighting makes it easier to compress the description lengths for attributes, and thus easier to find local optima with the proposed greedy algorithm.

Twitter. We consider a network created by interactions among Twitter users on the subject related to Proposition 30 on the November 2012 California ballot [Smith et al.

2013b]. We used the method described in Smith et al. [2013b] 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. This was done to deemphasize attributes common to all nodes, such as "#prop30."

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 [McAuley and Leskovec 2012]. 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 [McAuley and Leskovec 2012], with some users being members of multiple circles and other users not in any circle.

ArnetMiner. The ArnetMiner dataset is a citation network [Tang et al. 2009], 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 [Sen et al. 2008] is a citation network with 3,312 scientific papers, classified into one of six classes, and 4,732 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 3,703 unique words.

Pubmed. The Pubmed Diabetes dataset [Sen et al. 2008] 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 [McAuley and Leskovec (2012)] 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, and so on. The attributes of a single node (image) include image features that are obtained from PASCAL [Everingham et al. 2015], ImageCLEF [Henning Mller and (Eds.) 2010], MIR [Huiskes and Lew 2008], and NUS-wide [Chua et al. 2009]. We use the ground truth labels in the image classification tasks as the ground truth communities (only around 10% nodes have ground truth communities).

5.1. Qualitative Evaluation

We first look at the Twitter interaction network for Proposition 30. Figure 3(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 different are the identified communities from the ground truth?

The ground truth for this network is shown in Figure 3(b), in which we placed nodes in the same locations, but colored them according to their stance, i.e., 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 holding one stance on the topic, and a

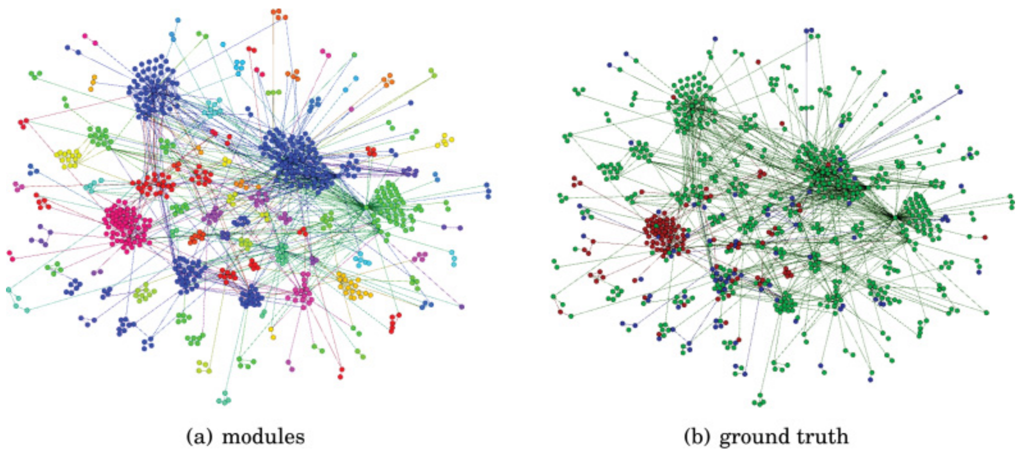


Fig. 3. Largest component in Proposition 30 user network. (a) Communities found by the Content Map Equation with bottom up algorithm, indicated by colors and locations. (b) Ground truth labels of the users in the network in (a): for (green), against (red), neutral (blue) (best viewed in color.)

few smaller communities comprised of individuals with different stances. Although the CME breaks users into many communities, the communities themselves are relatively pure, i.e., composed of users who have the same stance.

Next, we visualize the communities found by CME in the ArnetMiner citations network. Note that both color and position are used together to differentiate communities found by the CME. For example, the red community within the *DB* box, though looking similar in color to *IR* community, is not the same one as *IR*. From Figure 4, we are able to observe that though CME identifies more communities than the ground truth number of topics, most of the communities are very pure. Here, we use blue box to group communities, each consisting of nodes on only one topic, and we use dashed black box to denote communities with a mixture of topics. For instance, though the *DB* topic is mapped to ten communities, seven of them only contain nodes on the *DB* topic. The remaining three small communities, are a mixture of *DB* and related topics, such as *DM*, *WM*, *WS*, and *SW*. For example, the black dashed box with red color contains not only single-topic nodes of both WS and *DB*, but also nodes that are members of both topics. One reason for this is that some topics co-occur very frequently in the ground truth (i.e., there exists a single node with multiple ground truth topic labels) and are very similar to one another. It is natural for CME to put these nodes together in the same community.

5.2. Quantitative Evaluation

We quantitatively evaluate network partitioning using the point-wise normalized F-measure, purity, and Normalized Mutual Information (NMI) 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 [Ruan et al. 2013] of the partitions P on the ground truth clustering G is then calculated as

$$F(P, G) = \sum_{p \in P} \left\{ \frac{|p|}{n} \max_{g \in G} F(p, g) \right\}. \tag{14}$$

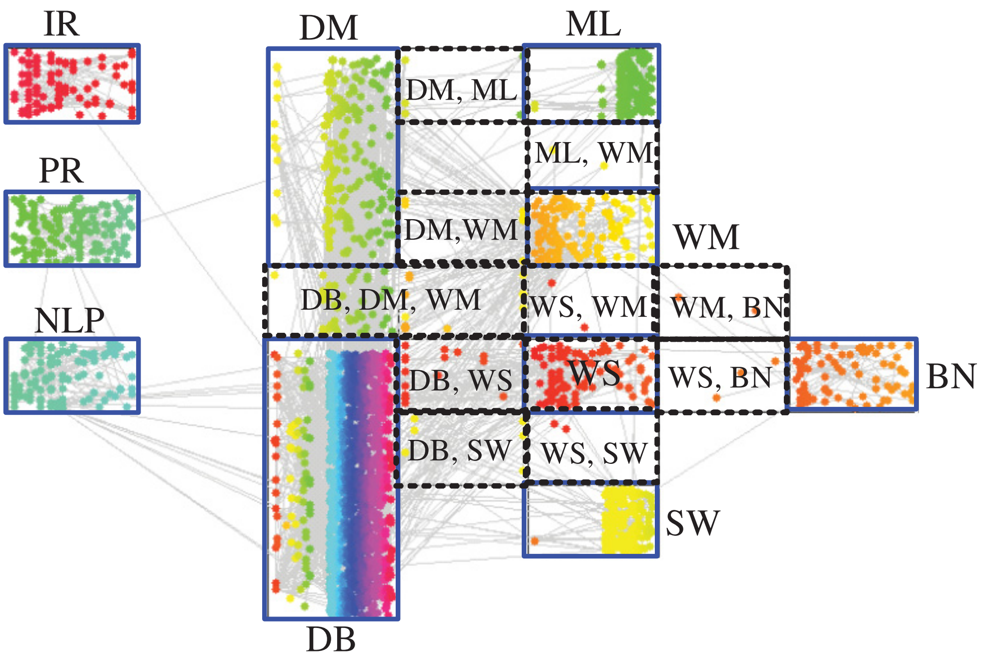


Fig. 4. ArnetMiner citation network. Blue squares indicate different topics, with nodes inside the squares identified solely with that topic. Nodes inside dashed squares are associated with multiple topics and are placed between both topics with a hyphenated label. Color and location differentiate nodes in different communities found by the CME with bottom-up algorithms (best viewed in color).

Purity. The purity of the partitions P on the ground truth clustering G is defined as

$$\texttt{purity}(P, G) = \texttt{avg}_{p\in P}\left\{\max_{g\in G}\frac{|p\cap g|}{|p|}\right\}. \tag{15}$$

Normalized Mutual Information. Given the identified clusters P and ground truth clustering G, the NMI is defined as

$$NMI(P, G) = \frac{2\times I(P;G)}{\max(H(P), H(G))},$$

where $H(P)$ and $H(G)$ denotes the entropy, and $I(P;G)$ is the mutual information between P and G, which is defined as

$$I(P;G) = \frac{\sum_i\sum_j p(p_i\cap g_j)\log\frac{p(p_i\cap g_j)}{p(p_i)p(g_j)}}{\sum_i\sum_j\frac{|p_i\cap g_j|}{n}\log\frac{n|p_i\cap g_j|}{|p_i||g_j|}}.$$

Baselines. We compare the proposed algorithms, the 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 [Blei et al. 2003]). Note that for LDA approach, each document is assigned with a distribution of topics (a set of overlapping clusters), whereas in the proposed methods, each node is assigned to a hard cluster. In order to achieve a fair comparison, based on the output of LDA, each node is assigned to the hard topic cluster with largest probability; (2) structure-based approaches such as the ME [Rosvall and Bergstrom 2008]; and (3) methods which use both links and attributes, such as BAGC [Xu et al. 2012] and Codicil [Ruan et al. 2013]. We do not compare to approaches [Yang et al. 2013; Zhu et al. 2011] since they only support a single-attribute per node. We do not include the comparison to baseline [Zhu

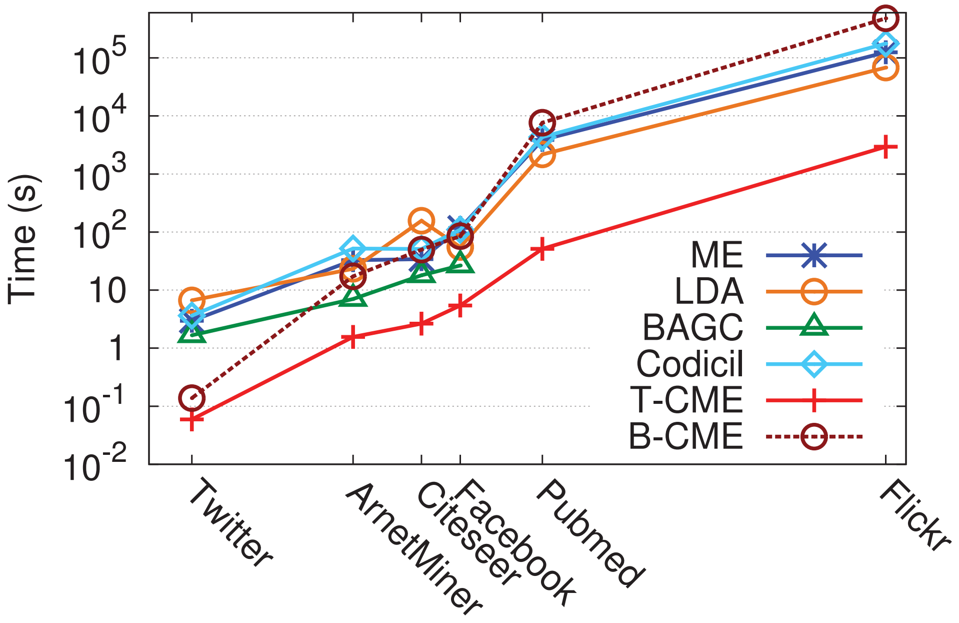


Fig. 5. Runtime comparison of different algorithms. Datasets are ordered by their size (best viewed in color).

et al. 2013] since we cannot obtain their implementation to reproduce the results. However, our method produces much better NMI scores on two benchmark datasets Citeseer and Pubmed. The experiments were performed on a 2.7GHz Intel i-7 CPU with 8G of memory.

5.2.1. Runtime. Figure 5 compares the runtimes of different methods. Results are ordered by network size (number of nodes and links, see Table IV), except the Citeseer citations network, which is between the ArnetMiner and Facebook datasets to improve visualization.

Note that for baseline BAGC, we only have results for the small to medium-size networks, since the implementation of BAGC 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, BAGC 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 the 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.

5.2.2. Performance. Figure 6 compares the F-measures obtained by different approaches. Recall that BAGC fails to run the Pubmed and Flickr datasets on our machine due to huge memory consumption.

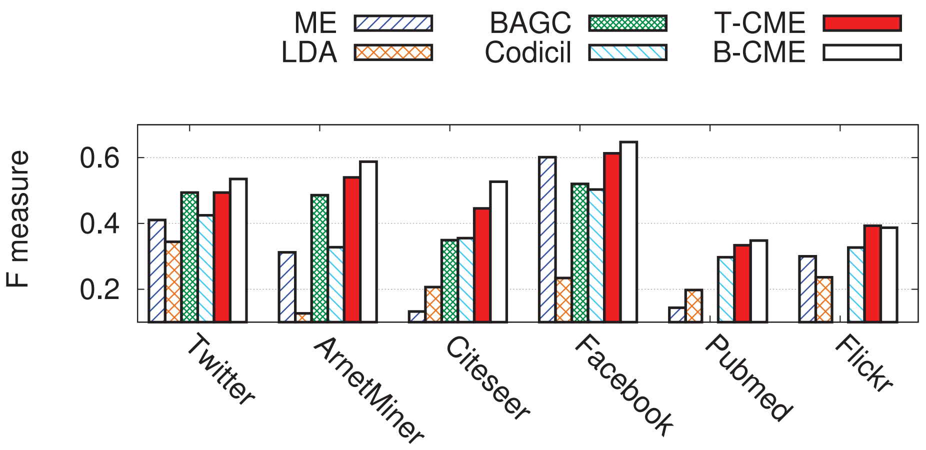


Fig. 6. Performance comparison in terms of F-measure (best viewed in color).

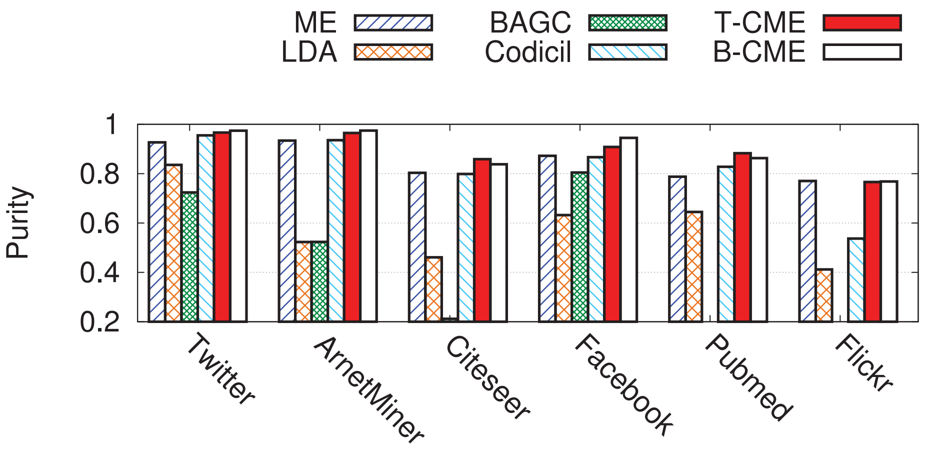


Fig. 7. Performance comparison in terms of purity (best viewed in color).

The results indicate that inclusion of node attributes leads 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 ME finds a worse grouping of papers than content-aware approaches. The CME is also much better than using content alone (e.g., LDA).

Compared to baselines BAGC and Codicil, the CME (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 tradeoff between efficiency and quality.

In addition to F-measure, we also use purity and NMI, shown in Figures 7 and 8, respectively, to evaluate network partitioning. Both results show that the CME outperforms the baseline BAGC 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. This is not surprising if we look at the description length of the partitions found by the two different search strategies (see Table V).

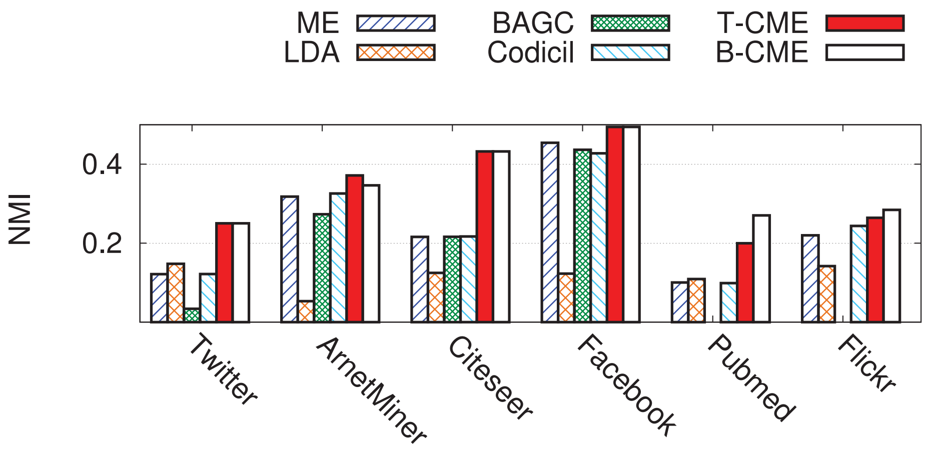


Fig. 8. Performance comparison in terms of NMI (best viewed in color).

Table V. Average Minimum Description Length (MDL)

MDL	Twitter	ArnetMiner	Citeseer	Facebook	Pubmed	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

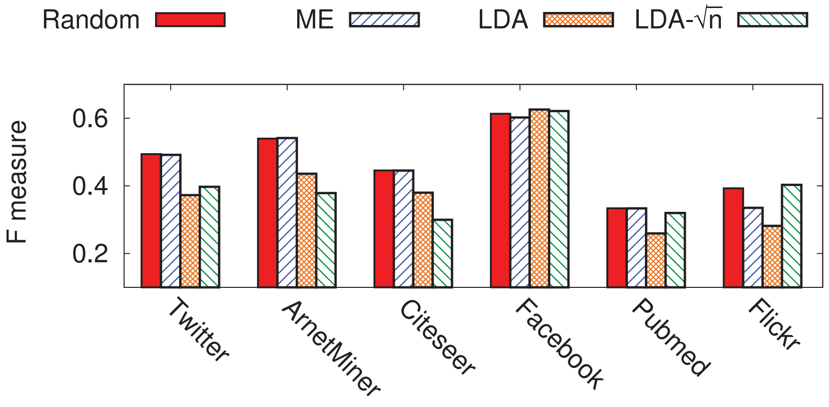
The T-CME achieves lower description length than B-CME for Citeseer and Pubmed as well. These results are consistent with the intuition that if we can correctly categorize the data (high purity within cluster), then the data can be described with the highest efficiency (i.e., using the minimum message length).

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

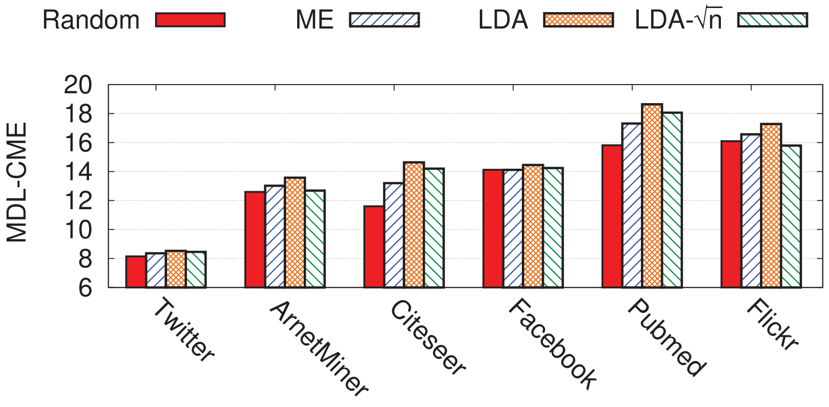
6. OPTIMIZATIONS

Having established the top-down method gives a good tradeoff between partition quality and runtime, we now investigate the effect of different optimizations of the top-down algorithm. Specifically, we look at 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 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 [Blei et al. 2003], to make the initial assignment. LDA requires the number of topics to be specified; hence, LDA-$\sqrt{n}$ means the number of topics is $\sqrt{n}$, and LDA means the number of topics is the true number of classes in the respective dataset. Alternatively, we can initialize the partition based on links alone, e.g., using the ME, and then use attributes to find a locally better solution with CME. We compare the partition quality resulting from random initialization to that resulting from LDA or ME.

Figure 9(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 five of six datasets, LDA-$\sqrt{n}$ in four of six datasets, and better than ME initialization in three of six datasets. Since the CME already incorporates content information equally with link information, the LDA/ME initializations only increase the contribution of content/link information, which deteriorates performance.



(a)



(b)

Fig. 9. Comparison of the impact of different initializations on (a) F-measure and (b) Minimum description length (MDL) or the number of bits required by the Content Map Equation (CME) to describe the network partition.

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 9(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, it is better to use random initialization in the top-down method.

A possible improvement to the top-down method might involve running it multiple times from random initial conditions (random restarts). Given how quickly the algorithm runs, taking the best results found over a constant number of runs could yield solutions of higher quality while still allowing for acceptable runtimes, and without changing the overall time complexity. We leave this as a direction for future investigation.

7. CONCLUSION

We have proposed and evaluated an information theoretic method for finding the modular structure of networks with node attributes. Building on the ME of Rosvall and Bergstrom [2008], we incorporate a new term that summarizes the contribution of the attributes to the description length of a random walk. By minimizing the resulting

CME, 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 CME results in a partition that is closer to the ground truth division than using links alone or using alternative methods that take attributes into consideration. Moreover, in contrast to other methods, our framework does not require ad-hoc 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 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.

A. APPENDIX

A.1. Proof to Proposition

It is sufficient to show $\Delta L = \Delta L^{\gamma}$ from two partitions where a single node changes modules and the rest remain fixed. Then, the result for any two partitions can be obtained by repeating this process of moving a single node at a time.

Consider the node σ, the only node that changes modules between the two partitions M_1 and M_2. We assume that in partition M_1, node σ is in module A, and in partition M_2, node σ is in module B. Then, using subscripts 1 and 2 to denote the partition, we have

$$\begin{aligned}\Delta L &= L_C(M_1) - L_C(M_2)\\ &= \left(q_{\curvearrowright 1} H(\mathcal{Q}_1) + \sum_{i=1}^{m_1} p^i_{\circlearrowright 1} H(\mathcal{P}^i_1) + \sum_{i=1}^{m_1} p^{(i)}_1 H(\mathcal{X}^i_1)\right)\\ &\quad - \left(q_{\curvearrowright 2} H(\mathcal{Q}_2) + \sum_{i=1}^{m_2} p^i_{\circlearrowright 2} H(\mathcal{P}^i_2) + \sum_{i=1}^{m_2} p^{(i)}_2 H(\mathcal{X}^i_2)\right)\\ \Delta L^{\gamma} &= L^{\gamma}_C(M_1) - L^{\gamma}_C(M_2)\\ &= \left(q_{\curvearrowright 1} H(\mathcal{Q}_1) + \sum_{i=1}^{m_1} p^i_{\circlearrowright 1} H(\mathcal{P}^i_1) + \sum_{i=1}^{m_1} p^{(i)}_1 H(\mathcal{X}^i_{1,\gamma})\right)\\ &\quad - \left(q_{\curvearrowright 2} H(\mathcal{Q}_2) + \sum_{i=1}^{m_2} p^i_{\circlearrowright 2} H(\mathcal{P}^i_2) + \sum_{i=1}^{m_2} p^{(i)}_2 H(\mathcal{X}^i_{2,\gamma})\right).\end{aligned}$$

Here, $\mathcal{X}^i_{1,\gamma}$ and $\mathcal{X}^i_{2,\gamma}$ in the equation for ΔL^{γ} indicate the feature vectors of Equation (9). The vector of attributes only contributes to the final term of the CME, giving

$$\begin{aligned}\Delta L - \Delta L^{\gamma} = &\left(\sum_{i=1}^{m_1} p^{(i)}_1 H(\mathcal{X}^i_1) - \sum_{i=1}^{m_2} p^{(i)}_2 H(\mathcal{X}^i_2)\right)\\ &- \left(\sum_{i=1}^{m_1} p^{(i)}_1 H(\mathcal{X}^i_{1,\gamma}) - \sum_{i=1}^{m_2} p^{(i)}_2 H(\mathcal{X}^i_{2,\gamma})\right)\end{aligned}$$

$$= \left(\sum_{i=A,B} p_1^{(i)} H(\mathcal{X}_1^i) - \sum_{i=A,B} p_2^{(i)} H(\mathcal{X}_2^i) \right)$$
$$- \left(\sum_{i=A,B} p_1^{(i)} H(\mathcal{X}_{1,\gamma}^i) - \sum_{i=A,B} p_2^{(i)} H(\mathcal{X}_{2,\gamma}^i) \right).$$

The last equality stems from the fact the modules agree everywhere except in modules A and B. Let

$$\begin{aligned} C &= \sum_{i=A,B} p_1^{(i)} H(\mathcal{X}_1^i) - \sum_{i=A,B} p_2^{(i)} H(\mathcal{X}_2^i) \\ &= p_1^{(A)} H(\chi_1^A) + p_1^{(B)} H(\chi_1^B) - p_2^{(A)} H(\chi_2^A) - p_2^{(B)} H(\chi_2^B). \end{aligned}$$

We now include the information about which module node σ falls in. To simplify notation, we consider σ as being a node separate from the set of nodes associated with either module A or module B, i.e., $\sum_{\alpha\in A}$ is a sum over all nodes in module A except σ, and $\sum_{\alpha\in A\bigcup\sigma}$ is a sum over all nodes in module A including node σ. Then,

$$\begin{aligned} C \; &= p_1^{(A)} \sum_{j\in d(A\bigcup\sigma)} \frac{\sum_{\alpha\in A\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}{p_1^{(A)}} \log\left(\frac{\sum_{\alpha\in A\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}{p_1^{(A)}}\right) \\ &+ p_1^{(B)} \sum_{j\in d(B)} \frac{\sum_{\alpha\in B}\left(p_\alpha x_j^\alpha\right)}{p_1^{(B)}} \log\left(\frac{\sum_{\alpha\in B}\left(p_\alpha x_j^\alpha\right)}{p_1^{(B)}}\right) \\ &- p_2^{(A)} \sum_{j\in d(A)} \frac{\sum_{\alpha\in A}\left(p_\alpha x_j^\alpha\right)}{p_2^{(A)}} \log\left(\frac{\sum_{\alpha\in A}\left(p_\alpha x_j^\alpha\right)}{p_2^{(A)}}\right) \\ &- p_2^{(B)} \sum_{j\in d(B\bigcup\sigma)} \frac{\sum_{\alpha\in B\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}{p_2^{(B)}} \log\left(\frac{\sum_{\alpha\in B\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}{p_2^{(B)}}\right), \end{aligned}$$

where $d(\Omega)$ is the set of attributes with at least one node in Ω having a positive attribute weight ($x_j^\alpha > 0$ for some $\alpha \in \Omega$). We define $d(\Omega_\gamma)$ to be this set of attributes according to Equation (9). First, assume that $p_1^{(A)}, p_1^{(B)}, p_2^{(A)}, p_2^{(B)} > 0$, meaning that both modules contain at least one other node than σ. Simplifying,

$$\begin{aligned} C \; &= p_\sigma \log\left(\frac{p_2^{(B)}}{p_1^{(A)}}\right) + \left(\sum_{\alpha\in A} p_\alpha\right) \log\left(\frac{p_2^{(A)}}{p_1^{(A)}}\right) \\ &+ \left(\sum_{\alpha\in B} p_\alpha\right) \log\left(\frac{p_2^{(B)}}{p_1^{(B)}}\right) \\ &+ \sum_{j\in d(A)} \left(\sum_{\alpha\in A} p_\alpha x_j^\alpha\right) \log\left(1 + \frac{p_\sigma x_j^\sigma}{\sum_{\alpha\in A} p_\alpha x_j^\alpha}\right) \end{aligned} \tag{16}$$

$$+\sum_{j\in d(B)}\left(\sum_{\alpha\in B}p_\alpha x_j^\alpha\right)\log\left(1+\frac{p_\sigma x_j^\sigma}{\sum_{\alpha\in B}p_\alpha x_j^\alpha}\right)$$
$$+\sum_{j\in d(\sigma)}\left(p_\sigma x_j^\sigma\right)\log\left(\frac{\sum_{\alpha\in A\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}{\sum_{\alpha\in B\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}\right).$$

The first two-line terms in Equation (16) do not depend on the attribute vectors. We repeat this process for

$$C_\gamma=\left(\sum_{i=A,B}p_1^{(i)}H(\mathcal{X}_{1,\gamma}^i)-\sum_{i=A,B}p_2^{(i)}H(\mathcal{X}_{2,\gamma}^i)\right),$$

simplifying to be

$$\begin{aligned}C_\gamma =&p_\sigma\log\left(\frac{p_2^{(B)}}{p_1^{(A)}}\right)+\left(\sum_{\alpha\in A}p_\alpha\right)\log\left(\frac{p_2^{(A)}}{p_1^{(A)}}\right)\\ &+\left(\sum_{\alpha\in B}p_\alpha\right)\log\left(\frac{p_2^{(B)}}{p_1^{(B)}}\right)\\ &+\sum_{k\in d(A_\gamma)}\left(\sum_{\alpha\in A}p_\alpha x_k^\alpha\right)\log\left(1+\frac{p_\sigma x_k^\sigma}{\sum_{\alpha\in A}p_\alpha x_k^\alpha}\right)\\ &+\sum_{k\in d(B_\gamma)}\left(\sum_{\alpha\in B}p_\alpha x_k^\alpha\right)\log\left(1+\frac{p_\sigma x_k^\sigma}{\sum_{\alpha\in B}p_\alpha x_k^\alpha}\right)\\ &+\sum_{k\in d(\sigma_\gamma)}\left(p_\sigma x_k^\sigma\right)\log\left(\frac{\sum_{\alpha\in A\bigcup\sigma}\left(p_\alpha x_k^\alpha\right)}{\sum_{\alpha\in B\bigcup\sigma}\left(p_\alpha x_k^\alpha\right)}\right).\end{aligned}\tag{17}$$

Recalling the definition for the attribute vectors from Equation (9), we can write

$$\sum_{k\in d(A_\gamma)}p_\alpha x_k^\alpha=\sum_{j\in d(A)}p_\alpha\left(\underbrace{\frac{x_j^\alpha}{\gamma}+\cdots+\frac{x_j^\alpha}{\gamma}}_{\gamma\text{ times}}\right)=\sum_{j\in d(A)}p_\alpha x_j^\alpha.$$

Similarly,

$$\begin{aligned}\sum_{k\in d(A_\gamma)}\left(\sum_{\alpha\in A}p_\alpha x_k^\alpha\right)\log\left(1+\frac{p_\sigma x_k^\sigma}{\sum_{\alpha\in A}p_\alpha x_k^\alpha}\right)&=\sum_{j\in d(A)}\left[\gamma\left(\sum_{\alpha\in A}p_\alpha\frac{x_j^\alpha}{\gamma}\right)\log\left(1+\frac{p_\sigma\frac{x_j^\sigma}{\gamma}}{\sum_{\alpha\in A}p_\alpha\frac{x_j^\alpha}{\gamma}}\right)\right]\\ &=\sum_{j\in d(A)}\left(\sum_{\alpha\in A}p_\alpha x_j^\alpha\right)\log\left(1+\frac{p_\sigma x_j^\sigma}{\sum_{\alpha\in A}p_\alpha x_j^\alpha}\right),\end{aligned}$$

and

$$\textstyle\sum_{k\in d(B_\gamma)}\left(\sum_{\alpha\in B}p_\alpha x_k^\alpha\right)\log\left(1+\frac{p_\sigma x_k^\sigma}{\sum_{\alpha\in B}p_\alpha x_k^\alpha}\right)=\sum_{j\in d(B)}\left(\sum_{\alpha\in B}p_\alpha x_j^\alpha\right)\log\left(1+\frac{p_\sigma x_j^\sigma}{\sum_{\alpha\in B}p_\alpha x_j^\alpha}\right).$$

For the last-line term in Equation (17), we can use a similar analysis to find

$$\sum_{k\in d(\sigma_\gamma)} \left(p_\sigma x_k^\sigma\right) \log\left(\frac{\sum_{\alpha\in A\bigcup\sigma}\left(p_\alpha x_k^\alpha\right)}{\sum_{\alpha\in B\bigcup\sigma}\left(p_\alpha x_k^\alpha\right)}\right) = \sum_{j\in d(\sigma)} \left(p_\sigma x_j^\sigma\right) \log\left(\frac{\sum_{\alpha\in A\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}{\sum_{\alpha\in B\bigcup\sigma}\left(p_\alpha x_j^\alpha\right)}\right).$$

Therefore, $C_\gamma = C$ holds. Thus,

$$\Delta L - \Delta L^\gamma = C - C_\gamma = C - C = 0.$$

Therefore, we have shown that

$$\Delta L = \Delta L^\gamma.$$

A similar argument can be used to show this holds if any of $p_1^{(A)}$, $p_1^{(B)}$, $p_2^{(A)}$, and $p_2^{(B)}$ are equal to zero. □

ACKNOWLEDGMENTS

The authors are grateful to the anonymous reviewers for insightful comments and suggestions that helped improve the quality and clarity of this paper.

REFERENCES

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

D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent Dirichlet allocation. *J. Mach. Learn. Res.* 3 (2003), 993–1022.

S. Brin and L. Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* 30 (1998), 107–117.

D. S. Choi, P. J. Wolfe, and E. M. Airoldi. 2012. Stochastic blockmodels with a growing number of classes. *Biometrika* 99, 2 (2012), 273–284.

T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Yan-Tao. Zheng. 2009. NUS-WIDE: A real-world web image database from national university of Singapore. In *CIVR*.

F. R. K. Chung. 1996. *Spectral Graph Theory*. CBMS Regional Conference Series in Mathematics, Vol. 92. American Mathematical Society. http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0821803158.

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

M. Everingham, S. M. Ali Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2015. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision* 111, 1 (2015), 98–136.

S. Fortunato. 2010. Community detection in graphs. *Phys. Rep.* 486 (Jan. 2010), 75–174.

S. Günnemann, I. Färber, S. Raubach, and T. Seidl. 2013. Spectral subspace clustering for graphs with feature vectors. In *2013 IEEE 13th International Conference on Data Mining*. 231–240.

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

T. Deselaers H. Mller, P. Clough and B. Caputo (Eds.). 2010. *Experimental Evaluation in Visual Information Retrieval*. The Information Retrieval Series, Vol. 32, Springer.

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

M. J. Huiskes and M. S. Lew. 2008. The MIR flickr retrieval evaluation. In *MIR*.

M. Jerrum and A. Sinclair. 1988. Conductance and the rapid mixing property for Markov chains: The approximation of permanent resolved. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC'88)*. ACM, New York, NY, USA, 235–244. http://dx.doi.org/10.1145/62212.62234

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

J. McAuley and J. Leskovec. 2012. Learning to discover social circles in ego networks. *NIPS* (2012).

J. McAuley and J. Leskovec. 2012. In *ECCV (4) (Lecture Notes in Computer Science)*. Springer, 828–841.

M. McPherson, L. Smith-Lovin, and J. M. Cook. 2001. Birds of a feather: Homophily in social networks. *Annu. Rev. Sociol.* 27 (2001), 415–444.

F. Moser, R. Colak, A. Rafiey, and M. Ester. 2009. Mining cohesive patterns from graphs with feature vectors. In *Proceedings of the SIAM International Conference on Data Mining*. 593–604.

M. E. J. Newman. 2006. Finding community structer in networks using the eigenvectors of matrices. *Phys. Rev. E* 74, 3 (2006).

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

E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabási. 2002. Hierarchical organization of modularity in metabolic networks. *Science* 297, 5586 (2002), 1551–1555.

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

M. Rosvall and Carl T. Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* 105, 4 (29 Jan. 2008), 1118–1123. DOI:http://dx.doi.org/10.1073/pnas.0706851105

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

P. Sen, G. M. Namata, M. Bilgic, L. Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Magazine* 29, 3 (2008), 93–106.

C. E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.* 27 (1948), 379–423.

A. Silva, W. Meira, Jr., and M. J. Zaki. 2012. Mining attribute-structure correlated patterns in large attributed graphs. *Proc. VLDB Endow.* 5, 5 (2012), 466–477.

L. M. Smith, K. Lerman, C. Garcia-Cardona, A. G. Percus, and R. Ghosh. 2013a. Spectral clustering with epidemic diffusion. *Phys. Rev. E* 88 (2013), 042813. DOI:http://dx.doi.org/10.1103/PhysRevE.88.042813

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

D. A. Spielman and S.-H. Teng. 2007. Spectral partitioning works: Planar graphs and finite element meshes. *Linear Algebr. Appl.* 421, 2–3 (March 2007), 284–305. DOI:http://dx.doi.org/10.1016/j.laa.2006.07.020

J. Tang, J. Sun, C. Wang, and Z. Yang. 2009. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2009).

J. Ugander and L. Backstrom. 2013. Balanced label propagation for partitioning massive graphs. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*. 507–516.

U. von Luxburg. 2007. A tutorial on spectral clustering. *Stat. Comput.* 17, 4 (1 Dec. 2007), 395–416. DOI:http://dx.doi.org/10.1007/s11222-007-9033-z

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

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

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

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

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

Y. Zhu, X. Yan, L. Getoor, and C. Moore. 2013. Scalable text and link analysis with mixed-topic link models. In *Proc. of KDD*.

Received June 2015; revised June 2016; accepted July 2016