Compression of Flow Can Reveal Overlapping-Module Organization in Networks

Alcides Viamontes Esquivel* and Martin Rosvall

Integrated Science Lab, Umeå University, Sweden
(Received 6 May 2011; published 29 December 2011)

To better understand the organization of overlapping modules in large networks with respect to flow, we introduce the map equation for overlapping modules. In this information-theoretic framework, we use the correspondence between compression and regularity detection. The generalized map equation measures how well we can compress a description of flow in the network when we partition it into modules with possible overlaps. When we minimize the generalized map equation over overlapping network partitions, we detect modules that capture flow and determine which nodes at the boundaries between modules should be classified in multiple modules and to what degree. With a novel greedy-search algorithm, we find that some networks, for example, the neural network of the nematode *Caenorhabditis elegans*, are best described by modules dominated by hard boundaries, but that others, for example, the sparse European-roads network, have an organization of highly overlapping modules.

Subject Areas: Complex Systems, Computational Physics, Interdisciplinary Physics

I. INTRODUCTION

To discern higher levels of organization in large social and biological networks [1–5], researchers have used hard-clustering algorithms to aggregate highly interconnected nodes into hard, nonoverlapping modules [6–8] because they have assumed that each node plays only a single modular role in a network. Recently, because researchers have realized that nodes can play many roles in a network, they have detected overlapping modules in networks using three approaches: a hard-clustering algorithm that is run multiple times [9,10], a local clustering method that generates independent and intersecting modules [11–14], and link clustering, which assigns boundary nodes to multiple modules [15–17]. However, all these approaches have limitations: The first and second approaches require several steps or tunable parameters to infer overlapping modules, and the third approach necessarily overlaps all neighboring modules. In this paper, we introduce a flow-based and information-theoretic approach to revealing the overlapping-module organization of large networks. From fundamental principles of information theory, we derive a single objective function and can deduce simultaneously: How many modules are in a network? Which nodes belong to which modules? And which nodes should belong to multiple modules and to what degree? If we were to assign nodes only to single modules, we would run the risk of breaking functional components of the network. However,

with possibly overlapping modules of highly interconnected nodes that effectively capture flow for long times, we can identify good candidates for functional components in the network [6].

We seek to simplify a network based on the patterns of flow through the system and use the flow to determine what role nodes on the boundaries between modules play. To resolve which boundary nodes should belong to multiple modules, we take advantage of the information-theoretic principle that all regularities in data can be used to compress the data. That is, efficient descriptions of the flow through the system must necessarily highlight important structures in the network with respect to the flow. For example, in Fig. 1(a), Keflavik airport in Reykjavik, Iceland, connects Europe and North America in the global air-traffic network. When we summarize the network in modules with long flow-persistence times, should Reykjavik belong to Europe, to North America, or to both? In our framework, the answer depends on the traffic flow. More precisely, Reykjavik's role in the network depends on how many passengers fly to Keflavik from Europe and from North America and to what degree they visit Iceland as tourists and immediately return back home vs to what degree they use Keflavik as a transit between continental Europe and North America. Were there little or no traffic from and to North America, then Europe and North America would naturally form two disjoint modules, with Reykjavik being in Europe. But with many passengers traveling to Reykjavik from both Europe and from North America, the character of the flow must determine Reykjavik's role in the network. If intercontinental-transit flow dominates, Europe and North America form an integrated flow system that we can efficiently summarize in a single module. If instead, returning-tourist flow dominates,

---
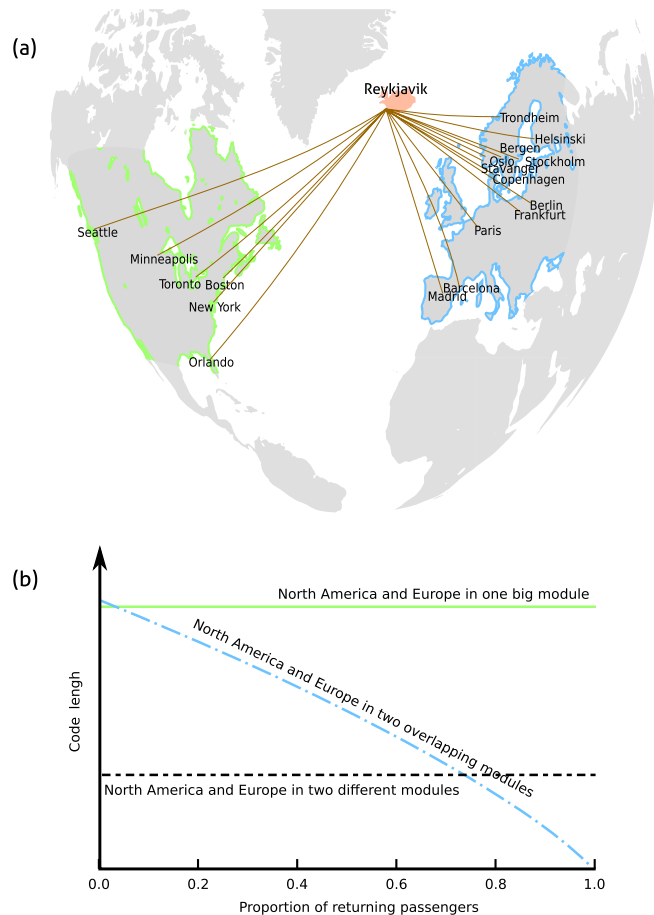
*a.viamontes.esquivel@physics.umu.se

FIG. 1. (a) Map showing that part of the global air-traffic network in which Keflavik airport in Reykjavik, Iceland, connects Europe and North America. The map equation for overlapping modules can exploit regularities in the boundary flow between modules. The three colored lines in (b) show the description length as a function of the proportion of returning passengers for three different partitions: North America, Europe, and Reykjavik in one big module (green solid line); North America and Europe in two nonoverlapping modules, with Reykjavik in either of the modules (black dashed line); and North America and Europe in two overlapping modules, with Reykjavik in both modules (blue dashed-dotted line).

Europe and North America form two separate flow systems, connected in Reykjavik, that we can more efficiently summarize in two overlapping modules. By generalizing the information-theoretic clustering method called the map equation [18,19] to overlapping structures, we can formalize this observation and use the level of compression of a modular description of the flow through the system to resolve which nodes should be assigned to multiple modules and form fuzzy boundaries between modules. With this approach, modules will overlap if they correspond to separate flow systems with shared nodes.

In the next section, we review the map-equation framework, introduce the map equation for overlapping modules, and explain how it exploits returning flow

near module boundaries. The mathematical framework works for both generalized and empirical flow, but in this paper we illustrate the method by exploring the overlapping-module structure of several real-world networks based on the probability flow of a random walker. We also test the performance on synthetic networks and compare the results with other clustering algorithms. Finally, in Sec. III, we provide complete descriptions of the map equation for overlapping modules and the novel search algorithm.

II. RESULTS AND DISCUSSION

A. The map equation

The mathematics of the map equation is designed to take advantage of regularities in the flow that connects the components of a system and generates their interdependence. The flow can be, for example, passengers traveling between airports, money transferred between banks, gossip exchanged among friends, people surfing the web, or, what we use here as a proxy for real flow, a random walker on a network guided by the (weighted, directed) links of the network. Specifically, the map equation measures how well different partitions of a network can be used to compress descriptions of flow on the network and utilizes the rationale of the minimum-description-length principle. We quote Peter Grünwald [20]: "...[E]very regularity in the data can be used to compress the data, i.e., to describe it using fewer symbols than the number of symbols needed to describe the data literally." In other words, the map equation gauges how successful different network partitions are at finding regularities in the flow on the network.

We employ two regularities for compressing flow on a network. First, we use short code words for nodes visited often and, by necessity, long code words for nodes visited rarely, such that the average code-word length will be as short as possible. Second, we use a two-level code for module movements and within-module movements, such that we can reuse short code words between modules with long persistence times.

Because we are not interested in the actual code words, but only in the theoretical limit of compression, we use Shannon's source-coding theorem [21], which establishes the Shannon entropy $H(\mathbf{p})$ as the lower limit of the average number of bits per code word necessary to encode a message, given the probability distribution $\mathbf{p}$ with probabilities p_1, p_2 of the code words,

$$H(\mathbf{p}) = -\sum_i p_i \log_2 p_i.$$

For example, if there is a message "ABABBAAB..." for which the symbols "A" and "B" occur randomly with the same frequency, that is, for which A and B are independent and identically distributed, the source-coding

theorem states that no binary language can describe the message with fewer than $-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$ bit per symbol. However, if A occurs twice as often as B, the regularity can be exploited and the message compressed to $-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} \approx 0.92$ bit per symbol. For flow on networks, we measure how many bits are needed, on average, for coding each step of the random walker by calculating the Shannon entropy based on the node-visit frequencies.

But flow or a random walker does not visit nodes independently. For example, if a network has a modular structure, once a random walker enters a tightly interconnected region in the network, in the next step she will most likely visit a node in the same tightly interconnected region, and she tends to stay in the region for a long time. To take advantage of this regularity and further compress the description of the walk, for each tightly interconnected region, we use a separate module codebook with an extra exit code that is used when the random walker exits the module, and an index codebook that is used after the exit code to specify which module codebook is to be used next. Now we can make use of higher-order structure in a network. For a modular network, we can describe flow on the network without ambiguities in fewer bits, using a two-level code, than we could do with only one codebook, because we use the index codebook only for movements between modules and can reuse short code words in the smaller module codebooks.

Given a network partition M, it is now straightforward to calculate the per-step-minimum-description length $L(\mathsf{M})$ of flow on the network. We use the Shannon entropy equation to calculate the average description length of each codebook and weight the average lengths by their rates of use. For a modular partition M with m modules, the map equation takes the following form:

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

For between-module movements, we use $q_{\curvearrowright}$ for the rate of use of the index codebook with module code words used according to the probability distribution Q. For within-module movements, we use $p_{\circlearrowright}^{i}$ for the rate of use of the i-th codebook with node and exit code words used according to the probability distribution $\mathcal{P}^i$.

By minimizing the map equation over network partitions, we can resolve how many modules we should use and which nodes should be in which modules to best capture the dynamics on the network. See the link given in Ref. [22] for a dynamic visualization of the mechanics of the map equation. Because the map equation depends only on the rates of node visits and module transitions, it is universal to all flow for which the rates of node visits and module transitions can be measured or calculated. The code structure of the map equation can also be generalized

to make use of higher-order structures. Reference [23] shows how a multilevel code structure can reveal hierarchical organization in networks, and in the next section, we show that we can capitalize on overlapping structures by relaxing the constraint that a node can belong only to one module codebook.

B. The map equation for overlapping modules

The code structure of the map-equation framework is flexible and can be modified to uncover different structures of a network as long as flow on the network can be unambiguously coded and decoded. As we will show here, by relaxing the constraint that a node can belong only to one module codebook and by allowing nodes to be *information free ports*, we can reveal the organization of overlapping modules in networks. To see how, let us again study the air traffic between North America and Europe in Fig. 1(a). Suppose that cities in North America and Europe belong to two different modules, which for simplicity are assumed to be identical in size and composition, and we assign membership to Reykjavik between North America and Europe. For a hard partition, we would assign Reykjavik to the module that most passengers travel to and from, and if the traffic flow were the same, we could choose either module. But if the flow to and from Reykjavik were dominated by American and European tourists visiting Iceland for sightseeing before returning to their home continent, both Americans and Europeans would consider Iceland as part of their territory. We can accommodate for this view if we allow nodes to belong to multiple module codebooks; depending on the origin of the flow, we use different code words for the same node.

With the map equation for overlapping modules, we can measure the description length of flow on the network with nodes assigned to multiple modules. By minimizing the map equation for overlapping modules, we can resolve not only how many modules a network is organized into and which nodes belong to which modules, but also which nodes belong to multiple modules and to what degree.

The pattern of flow—tourists returning to Iceland or businessmen in transit on intercontinental trips—determines whether we should assign Reykjavik to North America, Europe, or both. Or, conversely, when we decide whether Reykjavik should be assigned to North America, Europe, or both, we reveal the pattern of boundary flow between modules, as Fig. 1 illustrates. In this hypothetical example, assigning cities to two nonoverlapping modules is always better than assigning all cities to one module. But for a sufficiently high proportion of returning flow, the overlapping-module solution with Reykjavik in both modules as a free port provides the most efficient partition to describe flow on the network.

The map equation for overlapping modules can take advantage of regularities in the boundary flow between modules. To measure the length of an overlapping-module
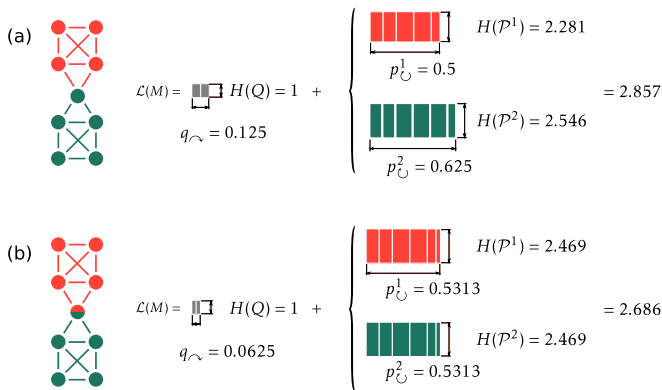
FIG. 2. The code structure of the map equation (a) without overlapping modules and (b) with overlapping modules. The color of a node in the networks at the left and of the corresponding block in the code structures at the right represents the module assignment; the width of a block in the code structures represents the node-visit rate; and the height H of the code-structure blocks represents the average length of code words in the codebooks.

description of flow on a network, we must decide how the flow switches modules in order to calculate the node-visit rates from different modules of multiply assigned nodes. In Sec. III, we provide a detailed description of how a random walker moves in an overlapping-module structure, but the rule is simple: When a random walker arrives at a node assigned to multiple modules, the walker remains in the same module if possible. Otherwise, the random walker switches, with equal probability, to one of the modules to which the node is assigned.

Figure 2 illustrates the code structure of a hard and a fuzzy partition of an example network with the dynamics derived from a random walker. For this network, the figure shows that an overlapping-module description allows us to describe the path of a random walker with fewer bits than we could do with a hard network partition. With overlapping modules, we halve the use of the index codebook, since the rate of module switching halves. Because we consequently use the exit codes in the now-identical module codebooks less often, the description of movements within modules also becomes shorter, even if the average code-word length increases. Turning the reasoning around again, given the overlapping-module organization, we have learned that returning flow characterizes the boundary flow between the modules.

With the mathematical foundation in place, we need an algorithm that can discover the best partition of the network. In particular, which nodes should belong to multiple modules and to what degree? For this optimization problem, we have developed a greedy-search algorithm that we call "fuzzy Infomap." (Details are given in Sec. III.) Here we give a short summary of fuzzy Infomap designed to provide good approximate solutions for large networks. We start from Infomap's hard clustering of the network [19] and then execute the two-step

algorithm. In the first step, we measure the change in the description length when we assign boundary nodes, one by one, to multiple modules. This calculation is fast, but aggregating the changes in the second step is expensive and often requires recalculating all node-visit rates. Therefore, we rank the individual multiple-module assignments and, in a greedy fashion, aggregate the individual best ones to minimize the description length.

C. Overlapping-module organization in real-world networks

To illustrate our flow-based approach, we have clustered a number of real-world networks based on the random-walk model of flow. Figure 3 shows researchers organized in overlapping research groups in the field of networks science. The underlying coauthorship network is derived from the reference lists in three review articles [1,6,24]. In this weighted and undirected network, we connect two researchers with a weighted link if they have coauthored one or more research papers. For every coauthored paper, we add to the total weight of the link a weight inversely proportional to the number of authors on the paper. Our premise is that two persons who have coauthored a paper have exchanged information, information that they can subsequently share with other researchers and induce a flow of information on the network. The map equation can capitalize on regularities in this flow, and Fig. 3 highlights one area of the coauthorship network with several overlapping research groups. For example, assigning Jure Leskovec to four research groups contributes to maximal compression of a description of a random walker on the network. Based on this coauthorship network, Leskovec is strongly associated with Dasgupta, Mahoney, Lang, and Backstrom, but also with groups at Cornell University, Carnegie Mellon University, Stanford University, and the Yahoo! Research organization. The size of the modules and the fraction of returning flow at the boundary nodes determine whether hard or fuzzy boundaries between research groups lead to optimal compression of flow on the network.

Table I shows the level of compression and overlap of a number of real-world networks. The networks are sorted from highest to lowest compression gain when allowing for overlaps. High compression corresponds to strong regularities (column C in Table I), and high compression gain for overlapping modules means that the network is characterized by interconnected flow systems (column ΔC_{fuzzy} in Table I). We find the highest compression gain in the European-roads network, which is a sparse network with road intersections as nodes and the roads themselves as links. Many intersections at boundaries between modules are classified in multiple modules, because intersections connect only a few roads and the return rate of the random flow is relatively high.

By contrast, compressing random flow in overlapping modules gives only a marginal gain over hard clustering in
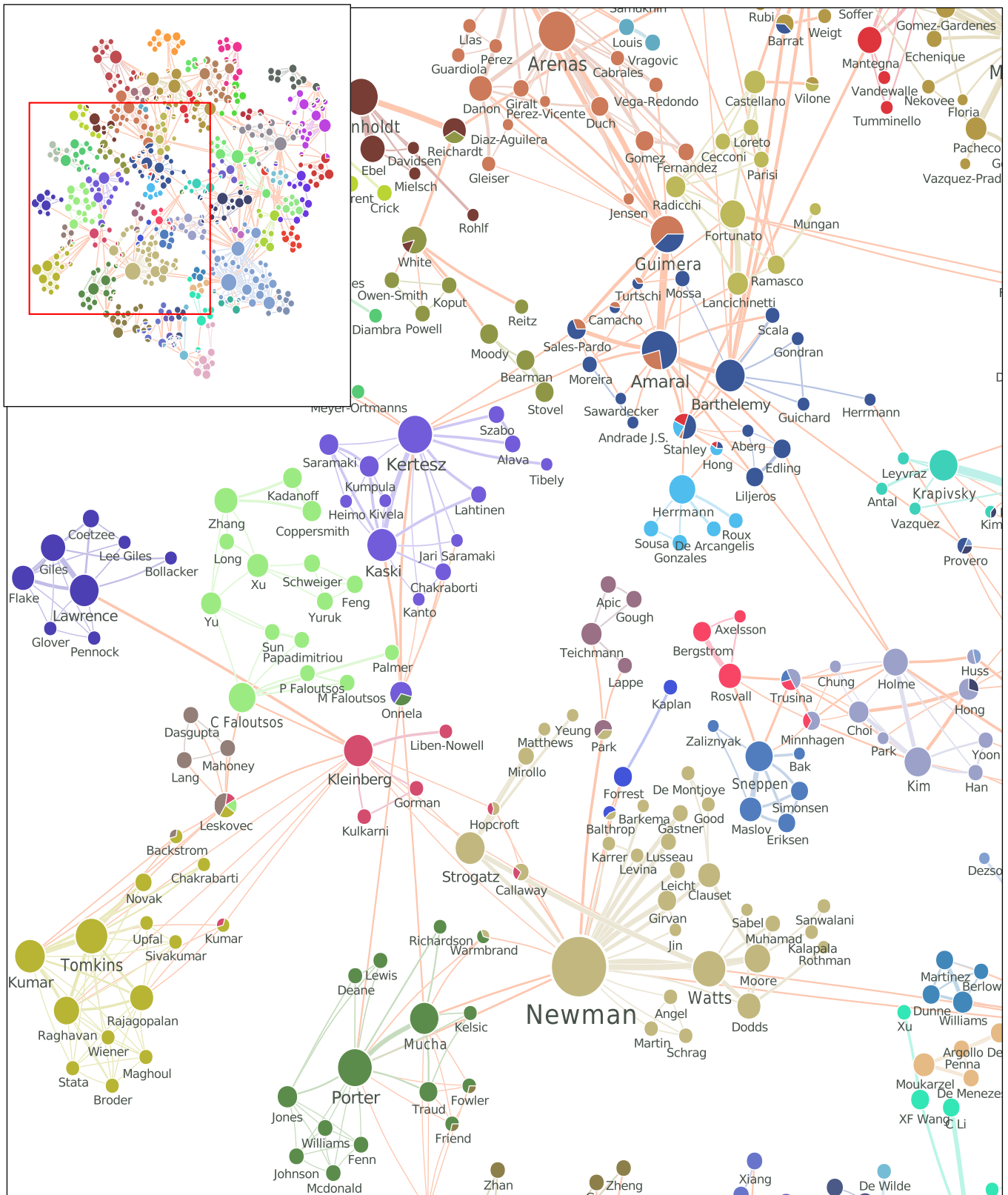
FIG. 3.   Network scientists organized in overlapping research groups. The colors of the nodes represent overlapping research groups identified by the map equation, the size of the nodes represents the importance of the researchers as quantified by the node-visit rates of the random walker; and the pie charts in certain nodes (for instance, for Amaral and Guimera in the upper right) represent the fractional association with the different research groups for researchers that have multiple assignments.

TABLE I.   The overlapping organization and the level of compression of eight real-world networks. For each network with n nodes and l links, we report the hard-partition compression C obtained with Infomap, the additional compression obtained with fuzzy Infomap, and the fraction of nodes that are assigned to multiple modules.

Network	n	l	C	ΔC_{fuzzy}	N_{fuzzy}/N
European-roads network [25]	1018	1274	46.2%	10.4%	35.5%
Western-states power grid [26]	4941	6994	53.4%	8.84%	27.5%
Human-diseases network [27]	516	1188	46.4%	2.87%	15.3%
Coauthorship network [28]	552	1317	48.9%	2.47%	14.6%
World air routes [29]	3618	14 142	31.1%	1.24%	13.9%
U.S. political blogs [30]	1222	16 714	4.13%	0.35%	5.81%
Swedish political blogs [31]	855	10 315	0.50%	0.18%	4.79%
Neural network of *C. elegans*[26]	297	2345	1.16%	0.13%	2.69%

the highly interconnected and directed neural network of the nematode *Caenorhabditis elegans*, in which network less than 3% of the neurons are classified in multiple modules. Even if there is evidence that the neural network is modular, we most likely underestimate the degree of overlap with a random-walk model of flow.

In the middle of the table, the world-air-routes network shows a relatively low compression gain, given the many cities classified in multiple modules. For this network, the compression gain would be much higher if, instead of random flow on the links, we were to describe real passenger flow with a higher return rate.

D. Comparing the map equation for overlapping modules with other methods

Depending on the system being studied and the research question at hand, researchers develop clustering algorithms for overlapping modules based on different principles. For example, some researchers take a statistical approach and see modules as nonrandom features of a network, while other researchers use a local definition and identify independent and intersecting modules, or take a link perspective and assign all boundary nodes to multiple modules. Consequently, the final partitions are quite different, and it is interesting to contrast our information-theoretic and flow-based approach, implemented in fuzzy Infomap, with these approaches, represented here by the OSLOM [13], clique-percolation [32], and link-communities [16] methods.

OSLOM defines a module as the set of nodes that maximizes a local statistical-significance metric. In other words, OSLOM identifies possibly overlapping modules that are unlikely to be found in a random network. Clique percolation identifies clusters by sliding fully connected k cliques to adjacent k cliques that share k-1 vertices with each other. A module is defined as the maximal set of nodes that can be visited in chained iterations of this operation, and the overlaps consist of the shared nodes between modules that do not support the slide operation across the boundary. Finally, the link-communities

approach creates highly overlapping modules by aggregating nodes that are part of a link community. The link communities themselves are built by hierarchical clustering based on a similarity score between links, the primal actors of the method.

To compare the methods at different degrees of overlap, we use a set of synthetic networks presented in Ref. [33]. In Table II, we include six statistics for the four methods applied to synthetic networks with 1000 nodes and three different degrees of overlap. (See Table II for details.) The first group of partition numbers describe the number of detected modules, the number of nodes that are assigned to multiple modules, and the total number of assignments. To interpret the results from a flow perspective, we include the index, module, and total code length for describing a random walker on the network for the given network partition.

Table II shows that fuzzy Infomap and OSLOM generate similar partitions for low and medium degrees of overlap, but the trend as the degree of overlap becomes higher indicates fundamental differences. By assigning boundary nodes to more modules than OSLOM prefers, fuzzy Infomap identifies modules with longer persistence times. The shorter index-code length resulting from the fewer transitions compensates for the longer module-code length from the larger modules. As a result, with the overlapping partitions generated by the fuzzy Infomap method, random flow can be described with fewer bits. But the difference is small and is seen only in the second decimal place when up to half of all the nodes are assigned to multiple modules.

Clique percolation generates partitions with more modules but fewer assignments than either fuzzy Infomap or OSLOM. From a flow perspective, smaller modules with less overlap give more module switches that cannot be compensated for by a shorter module-code length. The strength of the clique-percolation method is its simple definition that allows for easy interpretation of the results.

Designed with links as the primal actors used to identify pervasive overlap in networks, the results of the link-communities approach are quite different. For example,

TABLE II. Comparing four different methods for identifying overlapping clusters. We run fuzzy Infomap, OSLOM, and link-communities algorithms with their default settings and use clique-size four for the clique-percolation method. All values are averaged over ten instances of random undirected and unweighted networks with 1000 nodes and predefined community structure, generated with three different degrees of overlap [33]: Low-degree overlap corresponds to 100 nodes, medium-degree overlap corresponds to 300 nodes, and high-degree overlap corresponds to 500 nodes in multiple modules. All other parameters were held constant: The number of nodes to which multiply assigned nodes are assigned was set to two; each cluster consisted of on average 20 nodes with a minimum of 10 and a maximum of 50 nodes; and the power-law exponent was set to -2 for the node-degree distribution and -1 for the module-size distribution. Finally, the mixing parameter that controls the proportion of links within and between modules was set to 0.1.

	Partition numbers			Code length (bits)		
	Modules	Overlaps	Assignments	Index	Module	Total
Low-degree overlap						
Fuzzy Infomap	44	105	1228	1.7	5.9	7.6
OSLOM	44	89	1089	1.8	5.8	7.6
Clique percolation	43	104	1108	1.7	6.0	7.7
Link communities	3415	1000	9215	8.1	3.5	12
Medium-degree overlap						
Fuzzy Infomap	53	303	1830	2.2	6.0	8.2
OSLOM	54	276	1277	2.3	5.9	8.2
Clique percolation	55	268	1283	2.3	6.1	8.3
Link communities	4457	1000	11 628	8.7	3.5	14
High-degree overlap						
Fuzzy Infomap	56	398	1676	2.6	6.1	8.8
OSLOM	61	462	1465	2.8	6.0	8.8
Clique percolation	73	388	1429	2.9	6.1	9.0
Link communities	4298	1000	11 063	10	3.7	11

independent of the degree of overlap of the synthetic networks, each node belongs to, on average, ten modules. From the perspective of a random-flow model, the persistence time is short in the many small modules, and the information necessary to encode the many transitions is much larger than for the other methods. This result is expected, as the link-communities method is tailored to identify pervasive overlap in social networks in which people belong to several modules and information flow is far from random.

Often, performance is an important aspect to consider when choosing a clustering method. Therefore, we measured the time it took to cluster the synthetic networks with the different clustering algorithms. We stress that we used presumably nonoptimized research code made available online by its developers and that the performance depends, of course, on the network. For the 1000-node synthetic network used in our comparison, fuzzy Infomap used on average 1.7 seconds for a single iteration of module growth and 240 seconds for multiple growths, OSLOM used 330 seconds, the clique-percolation method 1.5 seconds, and link communities were identified in 2.4 seconds.

We conclude this comparison by stressing that the research question at hand must be considered when choosing a clustering method. Fuzzy Infomap provides fast results that, for a random-flow model, are similar to results generated by OSLOM and the clique-percolation method, at least for moderate degrees of overlap. On the other hand, for identifying pervasive overlap, researchers should consider the link-communities model or a generalized flow model with longer persistence times in smaller, highly overlapping modules.

III. MATERIALS AND METHODS

Here we detail the map equation for overlapping modules and describe our greedy-search algorithm.

A. The map equation for overlapping modules

In this section, we explain in detail how we derive the transition rates of a random walker between overlapping modules. We also derive the conditional probabilities for nodes assigned to multiple modules. We then express the map equation [Eq. (1)] in terms of these rates, which allows for fast updates in the search algorithm.

1. Movements between nodes assigned to multiple modules

To calculate the map equation for overlapping modules, we need the visit rates p_{α_i} for all modules $i \in M_\alpha$ to which a node α is assigned, and the inflow $q_{i\curvearrowright}$ and the outflow $q_{i\curvearrowleft}$ of all modules. We derive these quantities from the weighted and directed links $W_{\alpha\beta}$, which we normalize

such that $w_{\alpha\beta}$ correspond to the probability of the random walker moving to node β from node α:

$$w_{\alpha\beta} = \begin{cases} 0, & \text{if there is no link from } \alpha \text{ to } \beta, \\ \dfrac{W_{\alpha\beta}}{\beta \sum W_{\alpha\beta}}, & \text{otherwise.} \end{cases} \qquad (2)$$

When necessary, we use random "teleportation" or jumping to guarantee a unique steady-state distribution [34]. In other words, for directed networks, at rate τ, or whenever the random walker arrives at a node with no out-links, the random walker teleports to a random node in the network. To simplify the notation, we set $w_{\alpha\beta} = 1/n$ for all nodes α that have no out-links to all n nodes β in the network.

The movements between multiply assigned nodes and overlapping modules are straightforward. Whenever the random walker arrives at a node that is assigned to multiple modules, she remains in the same module if possible or switches to one of the other modules randomly if not possible. For example, assuming that the random walker is in module i, she remains in module i when moving to node β if node β is assigned to module i, $i \in M_\beta$. But if node β is not assigned to module i, $i \notin M_\beta$, she switches with equal probability $1/|M_\beta|$ to any of the modules to which node β is assigned (see Fig. 4). If we define the transition function

$$\delta_{\alpha_i\beta_j} = \begin{cases} 1, & \text{if } i = j, \\ \frac{1}{|M_\beta|}, & \text{if } i \neq j \text{ and } i \notin M_\beta, \\ 0, & \text{if } i \neq j \text{ and } i \in M_\beta, \end{cases} \qquad (3)$$

we can now define the visit rates by the equation system

$$p_{\alpha_i} = \sum_\beta \sum_{j \in M_\beta} p_{\beta_j} \delta_{\alpha_i\beta_j} \left[(1 - \tau) w_{\beta\alpha} + \tau \frac{1}{n} \right]. \qquad (4)$$

We solve for the unknown visit rates with the fast iterative algorithm BICGSTAB [35]. Since every node in module i guides a fraction $(1 - \tau)\sum_{\beta \notin i} w_{\alpha\beta}$ and teleports a fraction $\tau(1 - n_i/n)$ of its conditional probability p_{α_i} to nodes outside module i, the exit probability of module i is
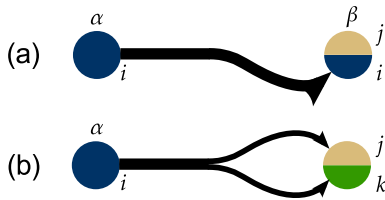

FIG. 4. Movements between nodes that are possibly assigned to multiple modules. (a) Assuming that the random walker is in module i, she remains in module i when moving to node β if node β is also assigned to module i. (b) However, if node β is not assigned to module i, she switches with equal probability to any of the modules to which node β is assigned.

$$q_{i\curvearrowright} = \sum_{\alpha \in i} p_{\alpha_i} \left[(1 - \tau) \sum_{\alpha \notin i} w_{\alpha\beta} + \tau \frac{n - n_i}{n} \right], \qquad (5)$$

where n_i is the number of nodes assigned to module i.

2. The expanded map equation for overlapping modules

To make explicit which terms must be updated in a given step of a search algorithm, here we expand the entropies of the map equation [Eq. (1)] in terms of the visit and transition rates p_{α_i}, $q_{i\curvearrowright}$, and $q_{i\curvearrowleft}$. When teleportation is included in the description length as above, the outflow of modules balances the inflow, but here we derive the outcome for the general case when $q_{i\curvearrowright} \neq q_{i\curvearrowleft}$.

We use the per-step probabilities of entering the modules $q_{i\curvearrowleft}$ to calculate the average code-word length of the index code words weighted by their rates of use, which is given by the entropy for the index codebook,

$$H(\mathcal{Q}) = -\sum_{i=1}^m \frac{q_{i\curvearrowleft}}{\sum_{j=1}^m q_{j\curvearrowleft}} \log_2 \frac{q_{i\curvearrowleft}}{\sum_{j=1}^m q_{j\curvearrowleft}}, \qquad (6)$$

where the sum runs over the m modules of the modular partition. The contribution to the average description length from the index codebook is the entropy $H(\mathcal{Q})$ weighted by its rate of use $q_\curvearrowleft$,

$$q_\curvearrowleft = \sum_{j=1}^m q_{j\curvearrowleft}. \qquad (7)$$

Substituting Eq. (7) into Eq. (6), we can express the contribution to the per-step average description length from the index codebook as

$$\begin{aligned} q_\curvearrowleft H(\mathcal{Q}) &= -q_\curvearrowleft \left[\sum_{i=1}^m \frac{q_{i\curvearrowleft}}{q_\curvearrowleft} \log_2 \frac{q_{i\curvearrowleft}}{q_\curvearrowleft} \right] \\ &= -\sum_{i=1}^m q_{i\curvearrowleft} [\log_2 q_{i\curvearrowleft} - \log_2 q_\curvearrowleft] \\ &= q_\curvearrowleft \log_2 q_\curvearrowleft - \sum_{i=1}^m q_{i\curvearrowleft} \log_2 q_{i\curvearrowleft}. \qquad (8) \end{aligned}$$

We use the per-step probabilities of exiting the modules $q_{i\curvearrowright}$ and the visit rates p_{α_i} to calculate the entropy of each module codebook:

$$\begin{aligned} H(\mathcal{P}^i) &= -\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta \in i} p_{\beta_i}} \log_2 \frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta \in i} p_{\beta_i}} \\ &\quad - \sum_{\alpha \in i} \frac{p_{\alpha_i}}{q_{i\curvearrowright} + \sum_{\beta \in i} p_{\beta_i}} \log_2 \frac{p_{\alpha_i}}{q_{i\curvearrowright} + \sum_{\beta \in i} p_{\beta_i}} \\ &= -\frac{1}{p_\cup^i} \left[q_{i\curvearrowright} \log_2 q_{i\curvearrowright} + \sum_{\alpha \in i} p_{\alpha_i} \log_2 p_{\alpha_i} - p_\cup^i \log_2 p_\cup^i \right], \qquad (9) \end{aligned}$$

with $p_\cup^i$ for the rate of use of the i-th module codebook,

$$p_{\cup}^i = q_{i\frown} + \sum_{\beta \in i} p_{\beta_i}. \qquad (10)$$

Finally, summing over all module codebooks, we obtain the description length given by the overlapping-module partition M:

$$L(\mathsf{M}) = q_{\frown} \log_2 q_{\frown} - \sum_{i=1}^{m} q_{i\frown} \log_2 q_{i\frown} - \sum_{i=1}^{m} q_{i\frown} \log_2 q_{i\frown}$$

$$- \sum_{i=1}^{m} \sum_{\alpha \in i} p_{\alpha_i} \log_2 p_{\alpha_i} + \sum_{i=1}^{m} p_{\cup}^i \log_2 p_{\cup}^i. \qquad (11)$$

The only visible difference between this expression and the map equation for nonoverlapping modules is the sum over conditional probabilities for nodes assigned to multiple modules, which is no longer independent of the overlapping-module partition M. But since the transition rates depend on the conditional probabilities [see Eq. (5)], all terms depend on the overlapping configuration.

B. The greedy-search algorithm for overlapping modules

To detect the overlapping-module organization of a network, ultimately we want to find the global minimum solution of the map equation over all possible overlapping-module configurations of the network, but only with an exhaustive enumeration of all possible solutions can we guarantee the optimal solution. This procedure is, of course, impractical for all but the smallest networks. However, we can construct an algorithm that finds a good approximation. Figure 5 explains the concept of our algorithm, which builds on an iterative two-step procedure.

In the first step, we individually assess which nodes are most likely to be assigned to multiple modules. Starting from a hard partition generated by Infomap [18] in the first iteration, we go through all nodes at the boundary between modules and assign each boundary node to adjacent modules. That is, one node and one adjacent module at a time, we assign the node to the adjacent module, such that the node is assigned to both its original modules and to the adjacent one, measure the map-equation change, and then return to the original configuration [see Fig. 5(c)]. Because the multiply assigned nodes connect only to singly assigned nodes in the first iteration, the conditional probabilities and the change in the map equation can be updated quickly without a full recalculation of the visit rates. This first step produces 3-tuples of local changes of the form *(node, extra-module, map-equation-change)*.

In the second step, we combine a fraction of all local changes generated in the first step into a global solution. Every time two or more multiply assigned nodes are connected, we need to solve a linear system to calculate the conditional probabilities. When a majority of nodes are
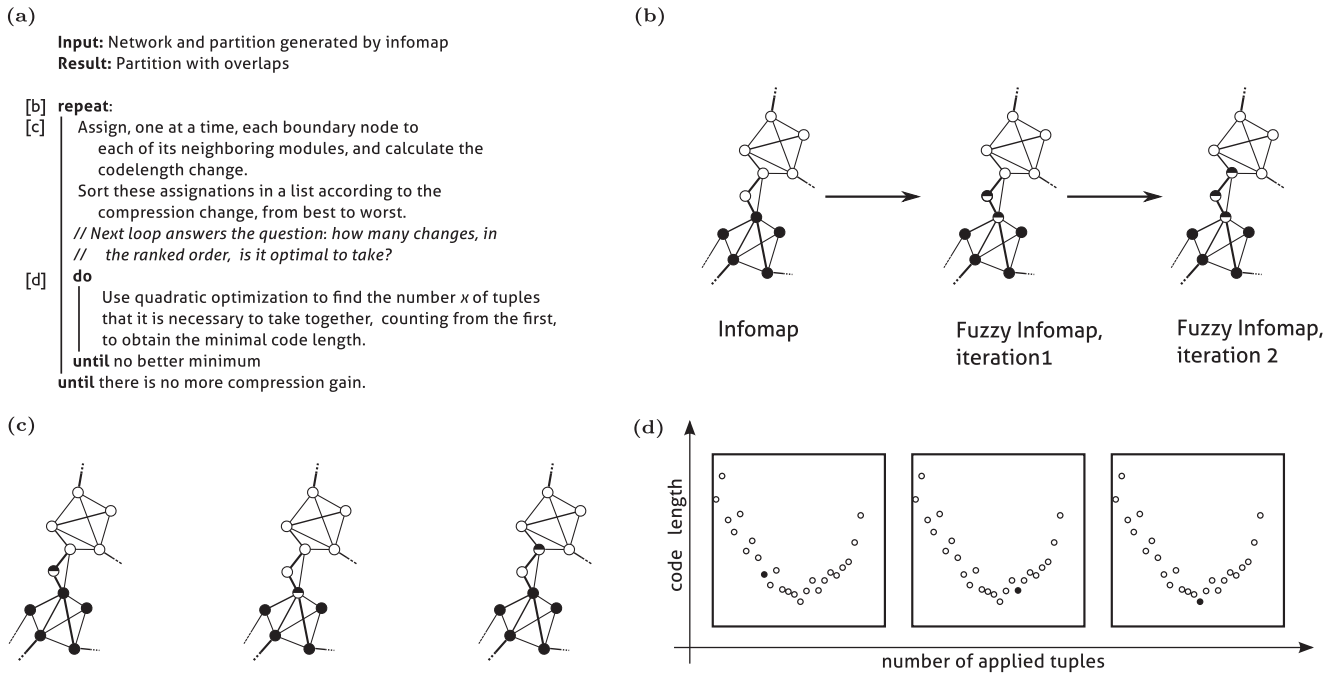

FIG. 5. General scheme of the two-step greedy-search algorithm for overlapping modules. (a) Pseudocode of the algorithm iteration [b] of first step [c] and second step [d]. Markers [b], [c], and [c] in the pseudocode correspond to figures (b), (c), and (d). Starting from a hard partition generated by Infomap [18], each iteration successively increases the overlap between modules to minimize the map equation for overlapping modules. In the first step (c), one by one, each boundary node is assigned to adjacent modules. In the second step (d), we first sort the local changes from best to worst and then iteratively apply quadratic fitting to find the number of best local changes that minimizes the map equation.

assigned to multiple modules, this can take as long as calculating the steady-state distribution of random walkers in the first place. For good performance, we therefore try to test as few combinations of local changes as possible. After testing several different approaches, we have opted for a heuristic method in which we first sort the tuples from best to worst in terms of map-equation change and then determine the number of best tuples that minimizes the map equation. The method works well, because good local changes often are good globally.

As a side remark, the map equation for the link-communities approach [17] allows for a straightforward and fast calculation of all conditional probabilities and transition rates, since each link belongs to only one module. However, this constraint enforces module switches between boundary nodes that belong to the same module, because all boundary nodes belong to multiple modules in the link-communities approach.

Figure 5(d) shows the value of the map equation as a function of the number of aggregated tuples ordered from best to worst. Combinations of tuples that individually generate longer description lengths can generate a shorter description length if they are applied together. This fact, together with the greedy order in which we aggregate the tuples, generates noise in the curve. To quickly approach the global minimum, we must overcome bad local minima caused by the noise and evaluate as few aggregations as possible. Therefore, we iteratively fit a quadratic polynomial to the curve by selecting new points at the minimum of the polynomial. A quadratic polynomial requires only three points to be fully specified, but in order to deal with the noise, we use a moving-local-least-squares fit. In practice, we evaluate around ten points for each quadratic fit and repeat this procedure a few times to obtain a good solution.

Step 1 and step 2 can now be repeated, each time starting from the obtained solution with overlapping modules from the previous iteration. Figure 5(b) illustrates that, by repeating the two steps, we can sometimes extend the overlap between modules, but this extension comes at a cost. After the first iteration of the algorithm, step 1 can also involve solving a linear system to calculate the conditional probabilities. Thus, the first step is no longer guaranteed to be as fast as in the first iteration. Still, for medium-sized networks, multiple iterations are feasible. For example, for the networks presented in Table I, the first iteration took a few seconds, and multiple iterations until the point of no further improvements took less than two minutes on a normal laptop. We have made the code available at the web site given in Ref. [36].

IV. CONCLUSIONS

To identify integrated structures in networks that capture flow for long times and that possibly extend over each other, we have introduced the map equation for overlapping modules. When we allow nodes to belong to multiple-module codebooks and minimize the map equation over possibly overlapping network partitions, we can determine which nodes belong to multiple modules and to what degree. Compared to hard partitions detected by the map equation, we have further compressed descriptions of a random walker on all tested real-world networks, and therefore have revealed more regularities in the flow on the networks. The modules will overlap if they correspond to separate flow systems that have shared nodes. We find the highest degree of overlapping-module organization in sparse-infrastructure networks, but this result depends on our random-walk model of flow.

We have contrasted our flow-based approach with other approaches on benchmark networks with planted overlapping communities. For a random-flow model, the map equation identifies much fewer modules than the link-communities method, and similar numbers of modules as the statistical OSLOM method and the heuristic clique-percolation method, but with boundary nodes assigned to more modules. At a speed on a par with the fastest methods, our novel greedy-search algorithm can capture the overlapping-flow systems of networks that are good candidates for having functional components. Since our mathematical framework is not limited to random flow, it would be interesting to compare our results with results derived from studies of empirical flow.

ACKNOWLEDGMENTS

[1] R. Albert and A. L. Barabási, *Statistical Mechanics of Complex Networks*, Rev. Mod. Phys. **74**, 47 (2002).

[2] R. Pastor-Satorras and A. Vespignani, *Evolution and Structure of the Internet: A Statistical Physics Approach* (Cambridge University Press, Cambridge, England, 2004), ISBN 0521826985.

[3] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. U. Hwang, *Complex Networks: Structure and Dynamics*, Phys. Rep. **424**, 175 (2006).

[4] C. Ratti, S. Sobolevsky, F. Calabrese, C. Andris, J. Reades, and O. Sporns, *Redrawing the Map of Great Britain from a Network of Human Interactions*, PLoS ONE **5**, e14248 (2010).

[5] R. D. Alba and G. Moore, *Elite Social Circles*, Sociological Methods & Research **7**, 167 (1978).

[6] S. Fortunato, *Community Detection in Graphs*, Phys. Rep. **486**, 75 (2010).

[7] M. E. J. Newman and M. Girvan, *Finding and Evaluating Community Structure in Networks*, Phys. Rev. E **69**, 026113 (2004).

[8] M. E. J. Newman, *Fast Algorithm for Detecting Community Structure in Networks*, Phys. Rev. E **69**, 066133 (2004).

[9] D. M. Wilkinson and B. A. Huberman, *A Method for Finding Communities of Related Genes*, Proc. Natl. Acad. Sci. U.S.A. **101**, 5241 (2004).

[10] D. Gfeller, J.-C. Chappelier, and P. De Los Rios, *Finding Instabilities in the Community Structure of Complex Networks*, Phys. Rev. E **72**, 056135 (2005).

[11] G. Palla, A. L. Barabási, and T. Vicsek, *Quantifying Social Group Evolution*, Nature (London) **446**, 664 (2007).

[12] S. Gregory, in *Machine Learning and Knowledge Discovery in Databases: Proceedings, Part 1, European Conference, Antwerp, 2008*, Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence, Vol. 5211, edited by W. Daelemans and K. Morik (Springer, New York, 2008), pp. 408–423.

[13] Andrea Lancichinetti, Filippo Radicchi, José J. Ramasco, and Santo Fortunato, *Finding Statistically Significant Communities in Networks*, PLoS ONE **6**, e18961 (2011).

[14] J. Whitney, J. Koh, M. Costanzo, G. Brown, C. Boone, and M. Brudno, in *Proceedings of the 11th International Conference on Algorithms in Bioinformatics*. Lecture Notes in Computer Science: Lecture Notes in Bioinformatics, edited by T. M. Przytycka and M.-F. Sagot (Springer-Verlag, Berlin, 2011), pp. 326–338.

[15] T. S. Evans and R. Lambiotte, *Line Graphs, Link Partitions, and Overlapping Communities*, Phys. Rev. E **80**, 016105 (2009).

[16] Yong-Yeol Ahn, James P. Bagrow, and Sune Lehmann, *Link Communities Reveal Multiscale Complexity in Networks*, Nature (London) **466**, 761 (2010).

[17] Y. Kim and H. Jeong, *Map Equation for Link Communities*, Phys. Rev. E **84**. 026110 (2011).

[18] M. Rosvall, D. Axelsson, and Carl T. Bergstrom, *The Map Equation*, Eur. Phys. J. Special Topics **178**, 13 (2009).

[19] M. Rosvall and Carl T. Bergstrom, *Maps of Information Flow Reveal Community Structure in Complex Networks*, Proc. Natl. Acad. Sci. U.S.A. **105**, 1118 (2008).

[20] P. D. Grünwald, *The Minimum Description Length Principle* (MIT, Cambridge, MA, 2007), ISBN 0262072815.

[21] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication* (University of Illinois, Champaign, IL, 1949).

[22] See http://www.mapequation.org.

[23] Martin Rosvall and Carl T. Bergstrom, *Multilevel Compression of Random Walks on Networks Reveals Hierarchical Organization in Large Integrated Systems*, PLoS ONE **6**, e18209 (2011).

[24] M. E. J. Newman, *The Structure and Function of Complex Networks*, SIAM Rev. **45**, 167 (2003).

[25] We compiled the road-network data from this source in 2010: http://europe.aaroads.com/eroads/erdlist.htm.

[26] D. J. Watts and S. H. Strogatz, *Collective Dynamics of 'Small-World' Networks*, Nature (London) **393**, 440 (1998).

[27] K. I. Goh, M. E. Cusick, D. Valle, B. Childs, M. Vidal, and A. L. Barabási, *The Human Disease Network*, Proc. Natl. Acad. Sci. U.S.A. **104**, 8685 (2007).

[28] We compiled the network in 2010 based on the coauthorships in Refs. .

[29] R. Guimera, S. Mossa, A. Turtschi, and L. A. N. Amaral, *The Worldwide Air Transportation Network: Anomalous Centrality, Community Structure, and Cities' Global Roles*, Proc. Natl. Acad. Sci. U.S.A. **102**, 7794 (2005).

[30] L. A. Adamic and N. Glance, in *Proceedings of the 3rd International Workshop on Link Discovery, 2005* (Association for Computing Machinery, New York, 2005), pp. 36–43, ISBN 1595932151 [http://dl.acm.org/citation.cfm?id=1134271].

[31] We compiled the network in 2010 from data provided by http://www.Twingly.com.

[32] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, *Uncovering the Overlapping Community Structure of Complex Networks in Nature and Society*, Nature (London) **435**, 814 (2005).

[33] A. Lancichinetti and S. Fortunato, *Benchmarks for Testing Community Detection Algorithms on Directed and Weighted Graphs with Overlapping Communities*, Phys. Rev. E **80**, 016118 (2009).

[34] S. Brin and L. Page, *The Anatomy of a Large-Scale Hypertextual Web Search Engine*, Computer Networks and ISDN Systems **30**, 107 (1998).

[35] H. A. van der Vorst, *Bi-cgstab: A Fast and Smoothly Converging Variant of bi-cg for the Solution of Nonsymmetric Linear Systems*, SIAM J. Sci. Comput. **13**, 631 (1991).

[36] See https://sites.google.com/site/alcidesve82/ for the code.