Data Mining and Knowledge Discovery (2022) 36:739–780
https://doi.org/10.1007/s10618-021-00809-w

Synwalk: community detection via random walk modelling

Christian Toth[1] · **Denis Helic**[2] · **Bernhard C. Geiger**[3]

Received: 22 January 2021 / Accepted: 15 October 2021 / Published online: 10 January 2022

Abstract

Complex systems, abstractly represented as networks, are ubiquitous in everyday life. Analyzing and understanding these systems requires, among others, tools for community detection. As no single best community detection algorithm can exist, robustness across a wide variety of problem settings is desirable. In this work, we present Synwalk, a random walk-based community detection method. Synwalk builds upon a solid theoretical basis and detects communities by synthesizing the random walk induced by the given network from a class of candidate random walks. We thoroughly validate the effectiveness of our approach on synthetic and empirical networks, respectively, and compare Synwalk's performance with the performance of Infomap and Walktrap (also random walk-based), Louvain (based on modularity maximization) and stochastic block model inference. Our results indicate that Synwalk performs robustly on networks with varying mixing parameters and degree distributions. We outperform Infomap on networks with high mixing parameter, and Infomap and Walktrap on networks with many small communities and low average degree. Our work has a potential to inspire further development of community detection via synthesis of random walks and we provide concrete ideas for future research.

Keywords Community detection · Clustering · Random walk modelling

Responsible editor: Annalisa Appice, Sergio Escalera, Jose A. Gamez, Heike Trautmann.

✉ Christian Toth
christian.toth@tugraz.at

Denis Helic
dhelic@tugraz.at

Bernhard C. Geiger
geiger@ieee.org

1 Signal Processing and Speech Communication Laboratory, Graz University of Technology, Inffeldgasse 16c, 8010 Graz, Austria

2 Institute of Interactive Systems and Data Science, Graz University of Technology, Inffeldgasse 16c/I, 8010 Graz, Austria

3 Know-Center GmbH, Inffeldgasse 13/6, 8010 Graz, Austria

1 Introduction

Large-scale systems of various kinds including social, informational or biological systems are pervasive in human life. Prominent examples of such systems are online social networks, the Internet, power grids or neural networks in the brain. Naturally, there is a strong interest in analyzing and understanding these systems, e.g., to estimate the effects of interventions on parts of the system, to alter or preserve their functionality, or to predict their future evolution. Commonly, such systems are abstracted as networks, where nodes represent the entities in the system, and links between nodes represent the (form of) interaction between the entities.

There are three basic necessities for a feasible empirical research of large networks: (i) identification of functional groups, frequently called *communities* (Girvan and Newman 2002; Radicchi et al. 2004), (ii) models for the interaction within and between these functional groups, (iii) visualization of large networks and their dynamics in human graspable form.

Community detection (Girvan and Newman 2002; Fortunato 2010; Fortunato and Hric 2016) is an established tool satisfying these three basic requirements, as the identification of functional groups in networks is often facilitated by implicitly or explicitly assuming specific interaction models and further allows to visualize the network at a more granular level. Consequently, researchers proposed numerous community detection algorithms in recent years (Girvan and Newman 2002; Clauset et al. 2004; Rosvall and Bergstrom 2008; Rosvall et al. 2009; Pons and Latapy 2005; Blondel et al. 2008; Raghavan et al. 2007; Reichardt and Bornholdt 2006; Peixoto 2014a).

This plethora of algorithms leaves us frequently wondering which is the "best" community detection algorithm for a given practical application? Typically, researchers compare algorithms based on their ability to identify ground truth communities in artificially generated benchmark networks (Orman and Labatut 2009; Yang et al. 2016) or ground truth extracted from node-metadata in empirical networks. However, Peel et al. (2017) recently showed that such an evaluation is more delicate: they provide a No Free Lunch theorem, stating that there can be no single best algorithm for all possible detection scenarios, and furthermore, community evaluation on empirical networks based on node meta-data is contestable in a general case. Hence, as different algorithms (potentially) uncover different structural aspects when applied to a given network, the choice of method depends, among other criteria, on the type of community one is looking for.

One prominent class of community detection methods characterizes communities based on random walks on a network, with Infomap (Rosvall and Bergstrom 2008; Rosvall et al. 2009) and Walktrap (Pons and Latapy 2005) being two popular representatives (see Sect. 2 for a compact description). Along the lines of the No Free Lunch theorem, both methods have their strengths and their weaknesses. Whereas Infomap accurately uncovers communities that are strongly connected internally (as characterized by the mixing parameter; see Sect. 3.1), it fails to do so for loosely connected communities (cf. Yang et al. 2016, Sect. 5.2). On the other hand, while Walktrap delivers reasonable results over a broader spectrum of the strength of the community structures, we find that its performance strongly depends on the degree distribution of the network. In addition, Walktrap requires a selection of a hyper-parameter that is

commonly chosen empirically. As one typically does not know about the community structure of a network a priori, it is unclear for practitioners how to select among these two random walk methods.

This raises an interesting question: can we combine the strengths of both methods to arrive at a community detection method that is more robust across a wider range of problem settings? In this paper, we tackle this question by presenting Synwalk—a community detection method where we model community properties by designing a synthetic random walk model. Specifically, Synwalk assumes a class of random walks with independent and identically distributed (i.i.d.) movements within and between candidate communities. It then simultaneously optimizes the distribution parameters of these i.i.d. movements (closed-form solution) and the candidate community structure (combinatorial optimization) such that the thus synthesized random walk resembles the random walk induced by the network under consideration. Due to the structure of the i.i.d. movements and the aim to synthesize an existing random walk, Synwalk thus shares ideas from both Infomap and stochastic block modelling (Sect. 4.2). We discuss the properties of the resulting Synwalk objective in Sect. 4.1 and thoroughly investigate and compare the behavior of Synwalk to Infomap and Walktrap, the Louvain method, and stochastic block model inference on generated benchmark graphs in Sect. 5.2. Furthermore, we illustrate the applicability of our method on empirical undirected networks with non-overlapping communities (Sect. 5.3).

In this work we present a novel instance of community detection via random walk modelling, which adapts the concept of (stochastic) block modelling to random walk-based community detection. At the same time, Synwalk combines the strengths of the popular random walk-based community detection algorithms Infomap and Walktrap, achieving more robust results across a range of generated and empirical networks without the need for hyper-parameter optimization. We believe that our method and results can initiate future theoretical and practical work to fully unlock the potential of synthetic random walk models for community detection by, e.g., (i) designing objective functions that enable robust detection of communities on specific classes of networks, (ii) designing random walk models tailored for detecting specific types of communities, or (iii) employing different notions of graph-induced random walks to discover different aspects/communities of a network.

2 Related work

Different approaches to community detection have been inspired by different definitions of communities (see Fortunato and Hric 2016 for an excellent survey). Accordingly, Rosvall et al. (2019) argue that different approaches to community detection can be categorized into four big groups, i.e., cut-based community detection, clustering, stochastic block modelling, and community detection based on network flows or random walks. We will now briefly summarize the concepts and approaches relevant for this work.

In a classical view, communities are densely connected subnetworks of a network that are well separated, which resonates with cut- or clustering-based community detection. This view takes the internal and external node degrees w.r.t. an assumed

community structure into account. Popular metrics for measuring the existence and strength of a community structure are the mixing parameter (Lancichinetti et al. 2008; Lancichinetti and Fortunato 2009a) and the modularity (Newman and Girvan 2004; Newman 2006). The mixing parameter μ of a node is the ratio between the number of links to nodes outside of its community and the total number of its links. The related quantity modularity compares the density of links within communities to links between communities and is used as an objective function for community detection (Clauset et al. 2004; Blondel et al. 2008).

While we will use the mixing parameter and modularity in setting up and evaluating our experiments in Sects. 5.2 and 5.3 , the method we propose falls into the category of random walk-based community detection methods. Random walks provide a simple proxy for diffusion processes describing the dynamics of a network. Here, the notion of a community is related to the average time a random walker spends within a certain subgroup of nodes of a network, and community detection becomes equivalent to finding an appropriate state space partition of the corresponding random walk.

This connection was utilized by Piccardi (2011), who proposed finding a community structure such that in the next time step the random walker stays within its current candidate community with high probability. The similar notion of Markov stability discussed by Lambiotte et al. (2014) has connections to modularity maximization and Infomap's map equation. Other approaches to the aggregation of random walks include non-negative matrix factorization of the random walk's transition probability matrix to obtain a low-rank representation (Ghasemi et al. 2020), spectral aggregation techniques (Zhang and Wang 2020), or information-theoretic approaches to Markov chain aggregation, cf. Amjad et al. (2020), Faccin et al. (2020) and Deng et al. (2011). All these approaches can, under appropriate circumstances and settings, be utilized to partition a network into overlapping or non-overlapping communities.

Hurley and Duriakova (2015, 2016) proposed an information-theoretic method for community detection that combines a random walk-based approach with (classic) block modelling. More specifically, the authors aim to find a candidate clustering of the network under investigation such that the random walk induced on this clustering is similar to an arbitrarily chosen target random walk, where similarity is measured by the Kullback–Leibler divergence and optimized using a Hartigan-style optimization procedure.

Similarly to Faccin et al. (2020) and Piccardi (2011)), the method proposed in Hurley and Duriakova (2015, 2016)) is predominantly based on modelling random walks on clusters. In contrast, Synwalk is based on the random walk induced by the network under investigation, i.e., a random walk on the network's nodes.

Another recent method proposed by Peixoto and Rosvall (2017) detects communities on possibly dynamic networks by combining elements from Markov aggregation and stochastic block model inference. Specifically, they try to co-cluster the states and preceding trajectories ("memories") of multiple realizations of a random walk with the aim of minimizing its description length. Similarly to our approach, this work assumes a synthetic random walk model. However, whereas they try to infer its parameters (i.e., its transition probabilities) jointly with the optimal (co-)clustering in a Bayesian approach, our method makes explicit assumptions about the synthetic

random walk's parameters and tries to find the optimal clustering by comparing the synthetic to a graph-induced random walk.

We close this section by reviewing two prominent examples for random walk-based community detection methods, Infomap (Rosvall and Bergstrom 2008; Rosvall et al. 2009) and Walktrap (Pons and Latapy 2005), against which we will compare our approach experimentally.

Assuming a certain clustering (i.e., a candidate community structure), Infomap encodes the movements of a random walker on a network with a two-level codebook scheme. Each cluster has its own codebook with codewords for each member node, plus a dedicated exit codeword. Additionally, there is a global index codebook with codewords for each cluster. Now, for every move of the random walker, Infomap records the codeword of the next node from the codebook of its containing cluster. Moreover, whenever the random walker changes clusters, Infomap records the exit codeword of the old cluster's codebook and the codeword of the new cluster from the index codebook before it records the new node. By minimizing the average description length of realizations of such a random walk, Infomap obtains a clustering that compactly describes the network dynamics and hence should fit the true community structure well. Notably, it is not necessary to actually simulate random walks, as the movements of the random walker are characterized by the network topology, allowing to compute the average description length via the map equation (Rosvall and Bergstrom 2008; Rosvall et al. 2009).

Walktrap formulates a random walk-based distance measure between clusters. Given a fixed number of steps, a random walker starting at a certain node will visit a neighboring node with a given probability. These probabilities hold information about how well two nodes are connected. Now, assuming two nodes are within the same community (i.e., well-connected), their probabilities to reach any other node within the network for a given number of steps should be similar. This observation yields a distance measure based on a weighted mean squared difference of such probabilities. Walktrap greedily merges nodes/clusters based on the described distance to arrive at a suitable clustering (Pons and Latapy 2005).

While Infomap and Walktrap predict clusterings by analyzing these random walks, our method predicts clusterings by synthesizing the network-induced random walk from a restricted class of candidate random walks. Searching for a proper random walk within this class makes our method robust across different network types. Additionally, being able to design the candidate class opens up possibilities for exploring alternative designs in future research.

3 Preliminaries

3.1 Networks and clusterings

Let $\mathcal{G} := (\mathcal{X}, E, W)$ be a weighted network with nodes $\mathcal{X} = \{1, \ldots, N\}$, links $E \subseteq \mathcal{X}^2$ and weight matrix W. The weight matrix is given by $W := [w_{\alpha\to\beta}]_{\alpha,\beta\in\mathcal{X}}$ where $w_{\alpha\to\beta} \geq 0$ denotes the weight of the link $(\alpha, \beta) \in E$ starting at node α and pointing at node β. (We use Greek letters to indicate nodes.) For an undirected network

we set $(\alpha, \beta) = (\beta, \alpha)$ and require that either $(\alpha, \beta) \in E$ or $(\beta, \alpha) \in E$ to avoid the double counting of edges. A set $C \subseteq \mathcal{X}$ is a clique if it is a complete subnetwork of $\mathcal{G}$, i.e., for any two distinct nodes $\alpha, \beta \in C$ there exists a connecting link $(\alpha, \beta) \in E$.

For an undirected network, the degree k_α of node α is the number of links connected to it. We denote the average degree of the network as $\bar{k}$. The network density ρ is defined as

$$\rho = 2 \cdot \frac{|E|}{|\mathcal{X}|(|\mathcal{X}| - 1)} = \frac{\bar{k}}{|\mathcal{X}| - 1}. \tag{1}$$

Consider a *clustering* $\mathcal{Y}$ of $\mathcal{X}$ into a set of K nonempty elements $\mathcal{Y}_i$, i.e., $\mathcal{Y} := \{\mathcal{Y}_i | i \in \mathcal{K}^\mathcal{Y}\}$ where $\mathcal{K}^\mathcal{Y} = \{1, \ldots, K\}$ denotes the index set of $\mathcal{Y}$. We index the elements of such a clustering by Roman letters and refer to them as *clusters* or *communities*. If the clusters are disjoint we call them *non-overlapping* and the clustering a *partition*. A partition induces a mapping function $m: \mathcal{X} \to \mathcal{K}^\mathcal{Y}$, mapping each node of $\mathcal{G}$ to the index of its containing cluster, i.e., $m(\alpha) = i$ iff $\alpha \in \mathcal{Y}_i$. For the remainder of this paper we assume all clusterings to be partitions.

For an undirected, unweighted network and a candidate clustering $\mathcal{Y}$, the mixing parameter of node α is defined as (Lancichinetti et al. 2008; Lancichinetti and Fortunato 2009a)

$$\mu(\alpha) = \frac{k_\alpha^{ext}}{k_\alpha} \tag{2}$$

where k_α^{ext} is the number of links between α and nodes outside of its community $\mathcal{Y}_{m(\alpha)}$. A cluster $\mathcal{Y}_i$ is a strong community (Radicchi et al. 2004) if for all of its nodes $\mu(\alpha) < 0.5$, but communities can be defined in a weak sense also for larger values (Lancichinetti and Fortunato 2009a). Similarly, for an undirected, unweighted network and a candidate clustering $\mathcal{Y}$, we can define the modularity of the clustering as (Newman and Girvan 2004; Newman 2006)

$$Q = \sum_{i \in \mathcal{K}^\mathcal{Y}} \frac{|E_i|}{|E|} - \left(\frac{1}{2|E|} \sum_{\alpha \in \mathcal{Y}_i} k_\alpha \right)^2 \tag{3}$$

where $|E_i|$ denotes then number of internal edges in cluster $\mathcal{Y}_i$. Brandes et al. (2008) showed that the modularity ranges from $-1/2$ to 1, with small values indicating weak community structures of the candidate clustering $\mathcal{Y}$.

3.2 Random walks

We consider random walks $\{X_t\}_{t \in \mathbb{N}}$ on the network $\mathcal{G}$, i.e., $\{X_t\}$ is a first-order Markov chain on $\mathcal{X}$. We assume that its stationary transition probability matrix

$P := [p_{\alpha\to\beta}]_{\alpha,\beta\in\mathcal{X}}$ is derived from the network's weight matrix W via

$$p_{\alpha\to\beta} = \frac{w_{\alpha\to\beta}}{\sum_{\beta'} w_{\alpha\to\beta'}}. \tag{4}$$

While there are other notions of random walks on networks, cf. Lambiotte et al. (2014) and Masuda et al. (2017) for an overview, we selected this formulation due its simplicity and connection to modularity maximization and Infomap's map equation, cf. the discussion around equations (6) and (34) in Lambiotte et al. (2014). We furthermore initialize the random walk with an invariant state distribution $p := [p_\alpha]_{\alpha\in\mathcal{X}}$ that satisfies

$$p_\beta = \sum_\alpha p_\alpha p_{\alpha\to\beta}. \tag{5}$$

We assume that the network is strongly connected, thus p is unique and positive.

Setting $Y_t := m(X_t)$ defines a stationary process $\{Y_t\}$ on the clusters. Specifically, the marginal and joint probabilities describing $\{Y_t\}$ are obtained as

$$p_i := \mathbb{P}(Y_t = i) = \sum_{\alpha\in i} p_\alpha, \quad i \in \mathcal{K}^{\mathcal{Y}} \tag{6a}$$

and

$$p_{i,j} := \mathbb{P}(Y_{t+1} = j, Y_t = i) = \sum_{\alpha\in i}\sum_{\beta\in j} p_\alpha p_{\alpha\to\beta}, \quad i, j \in \mathcal{K}^{\mathcal{Y}}. \tag{6b}$$

We further abbreviate $p_{\neg i} := 1 - p_i = \mathbb{P}(Y_t \neq i)$ for the marginal complement, $p_{i,\neg j} := p_i - p_{i,j} = \mathbb{P}(Y_{t+1} \neq j, Y_t = i)$ for the joint complement, and $p_{i\to j} := \mathbb{P}(Y_{t+1} = j | Y_t = i)$, respectively $p_{i\not\to j} := 1 - p_{i\to j} = \mathbb{P}(Y_{t+1} \neq j | Y_t = i)$ for the conditional and its complement.

3.3 Information theory

We make use of the following quantities from information theory that are well-described by Cover and Thomas (2006, Chapter 2). Let Y, Z denote random variables (RV), then we call $H(Z)$ the entropy of Z, $H(Y|Z)$ the conditional entropy of Y given Z and $I(Y;Z)$ the mutual information between Y and Z. Furthermore, let p and q denote discrete probability distributions over the same alphabet. Then we call $D(p\|q)$ the Kullback–Leibler divergence between p and q. If p, q are Bernoulli distributions i.e., $p = [p_1, 1-p_1]$ and $q = [q_1, 1-q_1]$, then we abbreviate $D(p_1\|q_1) := D(p\|q)$. Furthermore, let $P := [p_{\alpha\to\beta}]_{\alpha,\beta\in\mathcal{Z}}$ and $Q := [q_{\alpha\to\beta}]_{\alpha,\beta\in\mathcal{Z}}$ be transition probability matrices of equal size. The Kullback–Leibler divergence rate $\overline{D}(\cdot\|\cdot)$ between two

stationary Markov chains governed by P and Q is

$$\overline{D}(P\|Q) := \sum_{\alpha\in\mathcal{Z}}\sum_{\beta\in\mathcal{Z}} p_\alpha\, p_{\alpha\to\beta} \log \frac{p_{\alpha\to\beta}}{q_{\alpha\to\beta}} \tag{7}$$

given that the Markov chains are irreducible (Rached et al. 2004, Th. 1).

4 Community detection via random walk modelling

We now introduce the Synwalk objective, derive some of its properties, and discuss its relations to Infomap, stochastic block modelling, and model reduction techniques for random walks. For the sake of readability we defer proofs to Appendix A.

4.1 Derivation and properties of the Synwalk objective

Assume a network $\mathcal{G} = (\mathcal{X}, E, W)$ with an inherent community structure $\mathcal{Y}^{\text{true}}$. Consider further a random walker moving on $\mathcal{G}$ governed by the transition probability matrix P, which is derived from the weight matrix W. We refer to this random walker as the *network-induced* random walker, as its movements depend on the topology of $\mathcal{G}$ (i.e., implicitly its community structure). In the next step we design a *synthetic* random walker, governed by some transition probability matrix $Q^{\mathcal{Y}}$ which, in contrast to P, explicitly depends on some candidate partition $\mathcal{Y}$. In essence, our approach then aims to find a partition $\mathcal{Y}$ such that the synthetic random walker behaves (stochastically) as similarly to the network-induced walker as possible. Intuitively, the resulting partition will resemble the intrinsic partition $\mathcal{Y}^{\text{true}}$ very closely. We formalize this concept in the following.

The transition probability matrix that governs the synthetic random walker has a particular structure that depends on a candidate partition $\mathcal{Y}$. Specifically, suppose that at a given time step the synthetic random walker is at node α in cluster $\mathcal{Y}_i \in \mathcal{Y}$. We decide whether to leave or to stay in the current cluster $\mathcal{Y}_i$ in the next time step based on a cluster-specific Bernoulli distribution $[s_i, 1 - s_i]$. In case of a cluster change, we choose a new cluster $\mathcal{Y}_j \neq \mathcal{Y}_i$ according to a distribution over clusters $[u_i]_{i\in\mathcal{K}^{\mathcal{Y}}}$. Finally, we choose the next node β lying in the new cluster $\mathcal{Y}_j$ by a cluster-specific distribution over nodes $[r_\beta^j]_{\beta\in\mathcal{Y}_j}$ (note that $\mathcal{Y}_j = \mathcal{Y}_i$ if we stay in the current cluster). This particular structure yields the transition probability matrix $Q^{\mathcal{Y}} = [q_{\alpha\to\beta}]_{\alpha,\beta\in\mathcal{X}}$ where

$$q_{\alpha\to\beta} = \begin{cases} r_\beta^{m(\beta)} \cdot (1 - s_{m(\alpha)}), & m(\alpha) = m(\beta), \\ r_\beta^{m(\beta)} \cdot s_{m(\alpha)} \cdot \frac{u_{m(\beta)}}{1-u_{m(\alpha)}}, & \text{otherwise.} \end{cases} \tag{8}$$

Note that when switching clusters we have to normalize the distribution over clusters by $1 - u_{m(\alpha)} = \sum_{k\neq m(\alpha)} u_k$ since we exclude the current cluster $m(\alpha)$ as a choice.

The aim is now to find a candidate partition $\mathcal{Y}$ and corresponding parameters of (8) that maximize the similarity between the synthetic and the network-induced random

walk. We quantify this similarity via the Kullback–Leibler divergence rate $\overline{D}(P\|Q^{\mathcal{Y}})$, i.e., the lower $\overline{D}(P\|Q^{\mathcal{Y}})$, the more similar are P and $Q^{\mathcal{Y}}$, and the more likely it is that the synthetic random walker produces realizations of random walks that are also *typical* for the network-induced random walker (Kesidis and Walrand 1993). Hence, the optimal partition $\mathcal{Y}^*$ satisfies

$$\mathcal{Y}^* \in \arg\min_{\mathcal{Y}} \left[\min_{\{[r_\alpha^i]_{\alpha\in\mathcal{Y}_i},\ s_i,\ u_i\}_{i\in\mathcal{K}^{\mathcal{Y}}}} \overline{D}(P\|Q^{\mathcal{Y}}) \right]. \tag{9}$$

The cluster-specific distributions over nodes and the cluster-specific Bernoulli distributions minimizing (9) can be shown to be

$$r_\alpha^{i,*} = \frac{p_\alpha}{p_i} = \mathbb{P}(X_t = \alpha, Y_t = i) \tag{10}$$

$$s_i^* = p_{i\not\to i} = \mathbb{P}(Y_{t+1} \neq i | Y_t = i). \tag{11}$$

Regarding the distribution over clusters $[u_i]_{i\in\mathcal{K}^{\mathcal{Y}}}$ there exists no closed-form solution to the best of our knowledge. Nevertheless, by choosing

$$u_i = p_i = \mathbb{P}(Y_t = i) \tag{12}$$

as a sub-optimal solution we can relax the original optimization problem in (9) to arrive at (see Proposition 1 in Appendix A.1)

$$\mathcal{Y}^* \in \arg\max_{\mathcal{Y}} \sum_{i\in\mathcal{K}^{\mathcal{Y}}} p_i D(p_{i\to i}\|p_i). \tag{13}$$

We hence define the *Synwalk objective* as follows.

Definition 1 The Synwalk objective for a given partition $\mathcal{Y}$ is

$$\mathcal{J}(\mathcal{Y}) := \sum_{i\in\mathcal{K}^{\mathcal{Y}}} p_{i,i} \log \frac{p_{i\to i}}{p_i} + p_{i,\neg i} \log \frac{p_{i\not\to i}}{p_{\neg i}} = \sum_{i\in\mathcal{K}^{\mathcal{Y}}} p_i D(p_{i\to i}\|p_i) \tag{14}$$

As we show in Proposition 2 in Appendix A.2, $\mathcal{J}(\mathcal{Y})$ is bounded via

$$0 \leq \mathcal{J}(\mathcal{Y}) \leq I(Y_t; Y_{t-1}) \leq I(X_t; X_{t-1}). \tag{15}$$

Note that for a given candidate clustering $\mathcal{Y}$ all probabilities in (14) can be computed from the fixed transition probabilities and the invariant state distribution induced by a network's weight matrix according to Sect. 3.2. Hence, optimizing the Synwalk objective is a combinatorial problem over all possible clusterings of a given network. It is a common problem that the number of possible clusterings grows super-exponentially in the number of nodes and thus, an exact solution is intractable. We therefore employ a suitable search algorithm to find a near optimal clustering w.r.t. the Synwalk objective. See Sect. 5.1 and Appendix B for further details.

The following observation supports the rationale behind Synwalk's aptness as a community detection method. Consider an unweighted network of disconnected cliques. As we show in Appendix A.3, the Synwalk objective achieves its global maximum for a community structure identical to the clique structure of this network. Although isolated cliques are an unrealistic scenario for community detection, they carry the intuition of the concept of a community, i.e., strong internal and weak external connections. Synwalk's optimal behaviour in this idealized edge case theoretically grounds our strong experimental results in Sect. 5.2. For additional insights based on theoretical considerations and synthetic toy data we refer the reader to Toth (2020, Sections 3.2 & 5.1).

4.2 Relation to Infomap and (stochastic) block modelling

The design of our random walk model was inspired by Infomap's coding scheme. Recall Infomap's two-level codebook structure described in Sect. 2, i.e., the cluster codebooks with node and exit codewords, and the global index codebook. The distributions assembling the dynamics of our synthetic random walker in (8) correspond to these codebooks: (i) the cluster-specific distributions over nodes $[r_\alpha^i]_{\alpha\in\mathcal{Y}_i}$ correspond to the cluster codebooks, (ii) the cluster-specific Bernoulli distributions $\{s_i\}_{i\in\mathcal{K}^\mathcal{Y}}$ determining a cluster change correspond to the exit codewords, and (iii) the distribution over clusters $[u_i]_{i\in\mathcal{K}^\mathcal{Y}}$ corresponds to the index codebook.

Thus, while Infomap takes an *analytic* approach to community detection by applying the minimum description length principle with a specific codebook structure, Synwalk takes a *synthetic* approach by trying to mimic the network-induced random walk with our synthetic random walk model.

The definition of $Q^\mathcal{Y}$ in (8) and of the optimization problem (9) are reminiscent of stochastic block modelling under Kullback–Leibler divergence. The main difference is that in stochastic block modelling, one tries to infer model parameters—e.g., community structure, inter- and intra-community edge probabilities—such that the likelihood of a given graph is maximized. In other words, block modelling infers the parameters of a random graph model, i.e., a generative model from which graphs can be drawn, such that the likelihood of the graph under consideration is maximized. An essential point for stochastic block models is that these models have limited degrees of freedom, and that a good fit between the model and the graph is achieved by selecting an appropriate candidate clustering for the former. In contrast, Synwalk first transforms the graph under consideration to a random walk model, characterized by the transition probability matrix P. Then, the aim of Synwalk is to infer the parameters—i.e., the community structure and parameters of $Q^\mathcal{Y}$—of another random walk model such that the resulting random walk is "close" to the original one in a well-defined sense. Furthermore, it is essential that $Q^\mathcal{Y}$ has less degrees of freedom than P; while, for N nodes and K candidate clusters, P has $N(N-1)$ degrees of freedom, the degrees of freedom of $Q^\mathcal{Y}$ are limited to $K+(K-1)+(N-K)=N+K-1$. Thus, Synwalk can adequately be interpreted as an approach to "random walk modelling".

Finally, Hurley and Duriakova (2015, 2016) proposed a method that combines random walks on networks with (generalized) block modelling. While they consider

the network-induced random walk on clusters rather than on nodes, they also use the Kullback–Leibler divergence to measure the similarity with a target random walk and, thus, the fitness of the candidate clustering. For a specific target random walk it can be shown that their approach becomes equivalent to the goal of maximizing $I(Y_t; Y_{t-1})$ (Hurley and Duriakova 2015, Sec. III.A). The same cost function was also proposed by Deng et al. (2011) for Markov chain aggregation, where it was shown that the bipartition of states is related to spectral partition via the Fiedler vector. It is also a special case of the cost function $I(Y_t; Y_{t+T})$ proposed by Faccin et al. (2020), who showed that maximizing this quantity for $T = 1$ and a random walk on an unweighted and undirected network is equivalent to maximizing the likelihood of a degree-corrected stochastic block model. By assuming a less restrictive structure of the distribution over clusters in (8) the optimization problem in (9) becomes equivalent to maximizing $I(Y_t; Y_{t-1})$ over the possible clusterings (see Appendix A.2 for a concrete derivation). Thus, we can achieve this cost function by designing a suitable synthetic random walk model.

5 Experimental evaluation

In the following experiments we compare Synwalk to four well established community detection methods, namely, Infomap (Rosvall and Bergstrom 2008; Rosvall et al. 2009) and Walktrap (Pons and Latapy 2005) (both random walk-based), Louvain (Blondel et al. 2008) (based on modularity maximization), and stochastic block model (SBM) inference (Peixoto 2014a). The source code for reproducing these experiments can be found at https://github.com/synwalk/synwalk-analysis.

5.1 Implementations

To find a near optimal clustering w.r.t. our Synwalk objective we reuse Infomap's stochastic and recursive search algorithm (Rosvall and Bergstrom 2010, Appendix S1). See Appendix B for additional information about our implementation. The resulting framework used in the course of this work can be found at https://github.com/synwalk/synwalk.

For Walktrap and Louvain we use the implementations provided by igraph (Csardi and Nepusz 2006). Note that for Walktrap we assume a default value of $T = 4$ where T is the hyper-parameter describing the random walk length used to compute the node and cluster distances. We use GraphTool (Peixoto 2014b) for inferring a degree-corrected SBM for a given network. Hereafter we will refer to the SBM inference method simply as GraphTool. Unless otherwise noted, we use default parameters of these implementations. We use the same setup in both our experiments with LFR benchmark graphs and with empirical networks.

Table 1 Parameter setup for generating the LFR benchmark networks

Parameter	Description	Parameter set A	Parameter set B
N_c^{max}	Maximum community size	$0.2 \cdot N$	$0.1 \cdot N$
N_c^{min}	Minimum community size	$0.25 \cdot N_c^{max}$	10
k^{max}	Maximum node degree	$0.95 \cdot N_c^{max}$	$0.95 \cdot N_c^{max}$
$\overline{k}$	Average node degree	$\{15, 25, 50\}$	20
β	Community size distribution exponent	1.0	1.0
γ	Degree distribution exponent	2.0	2.0

5.2 Experiments on the LFR benchmark

To validate and compare the results of community detection methods it is common practice to evaluate their performance on benchmark networks (Yang et al. 2016; Fortunato and Hric 2016; Newman and Girvan 2004; Lancichinetti and Fortunato 2009b; Orman and Labatut 2009) where the ground truth community structure is known. The prevalent benchmark in more recent studies (Yang et al. 2016; Orman and Labatut 2009) is the LFR benchmark (Lancichinetti et al. 2008; Lancichinetti and Fortunato 2009a) and hence, we adopt it in our experiments. We generate the LFR benchmark networks with parameters as given in Table 1.

We employ the adjusted mutual information (AMI, Vinh et al. 2010) as a performance measure when comparing the partitions found by different community detection algorithms with the ground truth community structure. AMI values close to 1 indicate high similarity between the found partition and the ground truth, whereas a values around 0 reflect low similarity. Let $\mathcal{Y}^{\text{true}}$ denote the ground truth clustering and $\mathcal{Y}$ any predicted clustering, then the AMI is defined as

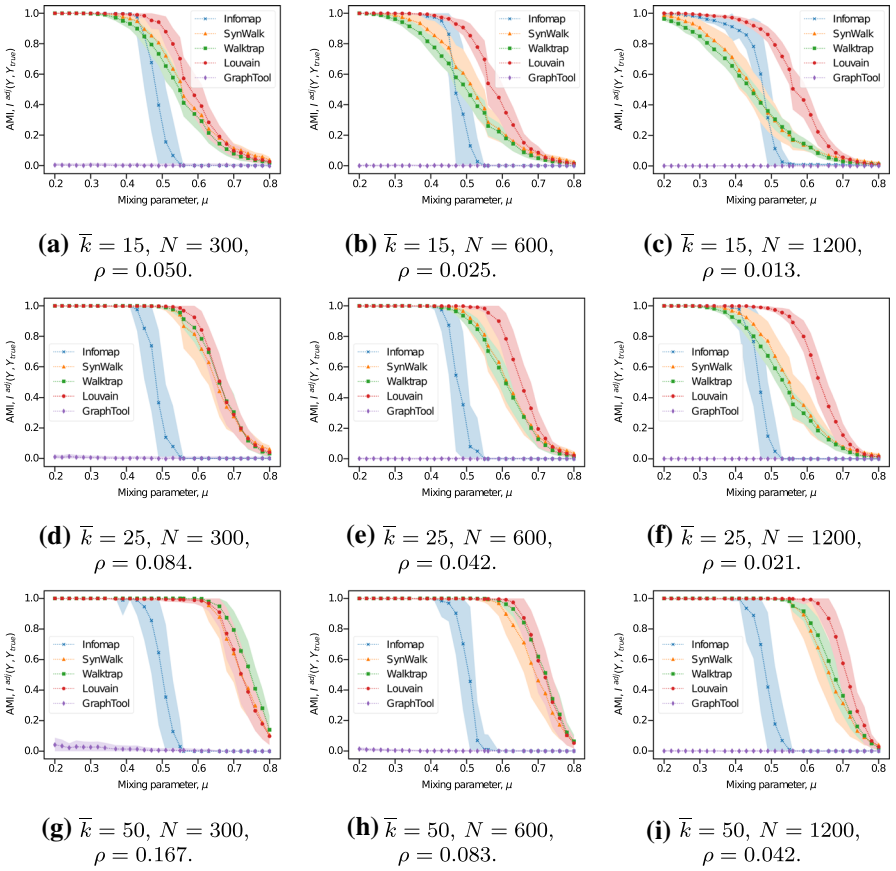
$$I^{adj}(\mathcal{Y}^{\text{true}}, \mathcal{Y}) = \frac{I(\mathcal{Y}^{\text{true}}; \mathcal{Y}) - \mathrm{E}\{I(\mathcal{Y}^{\text{true}}; \mathcal{Y})\}}{\frac{1}{2}[H(\mathcal{Y}^{\text{true}}) + H(\mathcal{Y})] - \mathrm{E}\{I(\mathcal{Y}^{\text{true}}; \mathcal{Y})\}}, \tag{16}$$

where $\mathrm{E}\{\cdot\}$ denotes the expectation operator with respect to a chosen permutation model. We normalize the AMI by the arithmetic mean as in (16).

5.2.1 AMI as a function of the mixing parameter

In this experiment we use parameter set A (see Table 1) to generate the LFR benchmark networks. We fix the network size and average degree of the generated LFR networks while varying their mixing parameter μ between 0.2 and 0.8. Experiments with varying network sizes are shown in Appendix C.

The results for the AMI as a function of the mixing parameter are shown in Fig. 1. As can be seen, Infomap correctly identifies the communities for sufficiently small values of μ and transitions to vanishing AMI around $\mu \approx 0.5$. This behavior reflects the definition of communities in a strong and weak sense as proposed by Radicchi et al. (2004) and was also observed by Yang et al. (2016). We explain this behaviour



(a) $\overline{k} = 15,\ N = 300,\ \rho = 0.050.$ **(b)** $\overline{k} = 15,\ N = 600,\ \rho = 0.025.$ **(c)** $\overline{k} = 15,\ N = 1200,\ \rho = 0.013.$

(d) $\overline{k} = 25,\ N = 300,\ \rho = 0.084.$ **(e)** $\overline{k} = 25,\ N = 600,\ \rho = 0.042.$ **(f)** $\overline{k} = 25,\ N = 1200,\ \rho = 0.021.$

(g) $\overline{k} = 50,\ N = 300,\ \rho = 0.167.$ **(h)** $\overline{k} = 50,\ N = 600,\ \rho = 0.083.$ **(i)** $\overline{k} = 50,\ N = 1200,\ \rho = 0.042.$

Fig. 1 Comparison of Infomap, Synwalk, Walktrap, Louvain and GraphTool on LFR benchmark networks with given average degree and network size. The lines and shaded areas show the mean and standard deviation of AMI as a function of the mixing parameter, obtained from 100 different network realizations. Synwalk outperforms Infomap for sufficiently high mixing parameter and network density. Performance of Synwalk and Walktrap increases with higher average degrees while holding the network density fixed (Color figure online)

by looking at Infomap's coding scheme. If $\mu > 0.5$, then the random walker will have a higher probability of exiting a community than staying within it. Hence, for the ground truth community structure, the coding overhead due to sending exit codewords will dominate, and clusterings resulting in more efficient encodings can be found, e.g., by putting all nodes into a single common cluster. Indeed, we observed exactly this behavior for Infomap in our experiments.

Unlike Infomap, Synwalk does not penalize frequent transitions between communities, although our random walk model resembles Infomap's coding scheme (cp. Sect. 4.2). Thus, the performance transitions of Synwalk, similarly to Walktrap, occur at increasing values of μ for increasing network densities (Fig. 1, columns from top to bottom and rows from right to left). Intriguingly, for roughly the same network density the transition phases shift to higher values of the mixing parameter as the

average degree increases (cp. Appendix C). Hence, neither the mixing parameter nor the network density sufficiently characterizes the AMI performance of Synwalk and Walktrap. We analyze this phenomenon further in Sect. 5.2.2. In contrast, our experiments show that even as we vary the average degree, Infomap's performance mainly depends on the mixing parameter of the networks.

Overall, Synwalk outperforms Infomap in terms of AMI on sufficiently dense networks or networks with mixing parameters $\mu \gtrapprox 0.5$. We perform approximately on par with Walktrap, where we see slightly better performance on networks with lower density (Fig. 1, top row) and a slight disadvantage on networks with higher density (Fig. 1, bottom row).

Considering the methods not based on random walks, we see that GraphTool does not perform well w.r.t. the AMI metric. We observed that GraphTool detects many small communities, apparently capturing a different aspect of the network structure. We think that inferring a hierarchical SBM may lead to better AMI values when looking at a clustering on a suitable hierarchy level (e.g. by choosing the clustering with the highest modularity score, similar to Walktrap).

Finally, Louvain (modularity maximization) yields the highest AMI values on networks with lower densities. For denser networks the performance of Synwalk and Walktrap comes close to or even slightly better than that of Louvain (cp. Fig. 1d, g, h).

We want to point out that the comparison between the different methods, more gravely between the random walk-based and non-random walk-based methods, should not solely be based on the AMI values achieved on this benchmark. For example, although GraphTool does not achieve good AMI performance on this benchmark, it certainly yields interesting results when applied to empirical networks (cf. Sect. 5.3). These results however will differ in their characteristics from methods based on other paradigms.

5.2.2 Classification analysis using node statistics

As we have seen in Sect. 5.2.1, the AMI performance of Synwalk and Walktrap on LFR networks transitions smoothly for varying values of the mixing parameter. To get deeper insights into the behavioral differences between Synwalk and Walktrap[1] we analyze the different qualities of their predictions in these transition phases.

For this purpose we analyze networks with varying network sizes and average degrees that are generated with parameter set A (see Table 1) while trying to keep the network density and the AMI (by appropriately setting the mixing parameter) constant (cp. main diagonal in Fig. 1). We align any predicted partitions to their respective ground truth partitions using a greedy matching algorithm as described in Appendix D. We then consider the nodes in the intersection of the ground truth communities with their aligned counterparts as correctly classified nodes, whereas the residual set of nodes form the group of misclassified nodes.

Given this distinction, we can compare the degree distributions of correctly classified and misclassified nodes. In addition to the node degree k_α, we consider the

[1] Due to the absence of a smooth transition phase for Infomap we do not analyze its behavior in this section.

normalized local degree (NLD) $\hat{k}_\alpha$, which we define as the ratio between the node degree and the maximum number of possible links in its containing cluster:

$$\hat{k}_\alpha = \frac{k_\alpha}{\binom{|\mathcal{Y}_{m(\alpha)}|}{2}} \quad \text{for} \quad |\mathcal{Y}_{m(\alpha)}| \geq 2. \tag{17}$$

Note that in general the NLD of a node will be different when computed w.r.t. its ground truth community or its predicted community.

The degree and NLD distributions are visualized in Figs. 2 and 3. Although in Figs. 1 and 10 in Appendix C we see an apparently strong dependence of the AMI performance on the average degree for Synwalk and Walktrap, for Synwalk a significant dependence is not visible in the class distributions (Fig. 2, top row). Nevertheless, the distributions of the NLDs w.r.t. the ground truth communities (Fig. 2, middle row) reveal that misclassified nodes are more likely to exhibit a low NLD than correctly classified ones.

In contrast, although the latter observation holds for Walktrap as well (Fig. 3, middle row), the node degrees of its misclassified nodes appear to be smaller than those of correctly classified nodes (Fig. 3, top row). This behavior appears plausible when considering the mechanics of Walktrap: nodes are grouped based on cluster/node distances that are computed by considering random walks of a specified length T (in our setup $T = 4$). For low values of T, low-degree nodes are rarely visited, resulting in frequent ties in distance calculations, whereas in the limit of $T \to \infty$ the distances are determined by the proportionality of the stationary distribution to the node degrees. Hence, it is necessary to make a trade-off regarding the random walk length T, which is typically chosen heuristically.

These differing properties of Synwalk and Walktrap manifest in contrasting detection behaviors on the LFR networks (cf. Fig. 4). Synwalk identifies smaller communities with greater accuracy than larger ones (dependence on the normalized local degree), i.e., the majority of misclassified nodes occur in the largest communities. While Walktrap follows this trend, misclassified nodes occur in smaller communities with increasing frequency (stronger dependence on node degree).

Another interesting difference appears when inspecting to which clusters misclassified nodes are assigned. Synwalk tends to place misclassified nodes in additional (i.e., clusters with no matching ground truth community), small clusters. Such behavior is indicated by the NLD distributions w.r.t. predicted communities as well, where misclassified nodes exhibit a significantly higher NLD than correctly classified ones (cp. Fig. 2, bottom row). This results in detected ground truth communities being ”pure”, i.e., they do not contain nodes from other ground truth communities. In contrast, Walktrap mainly confuses node memberships within clusters that do have a matching ground truth community. Again, these observations are supported by the NLD distributions w.r.t. predicted communities as well, where misclassified nodes exhibit a similar NLD to correctly classified ones (cp. Fig. 3, bottom row).

These behavioral differences between Synwalk and Walktrap are visible in the AMI performance as well: whereas both methods misclassify approximately the same

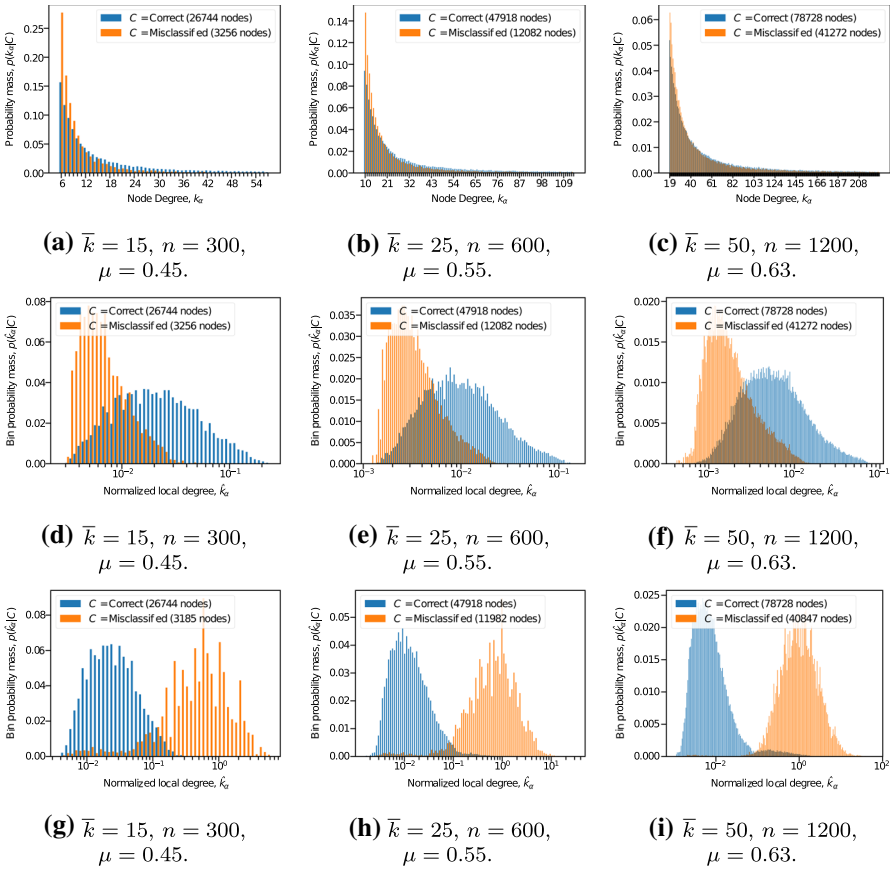
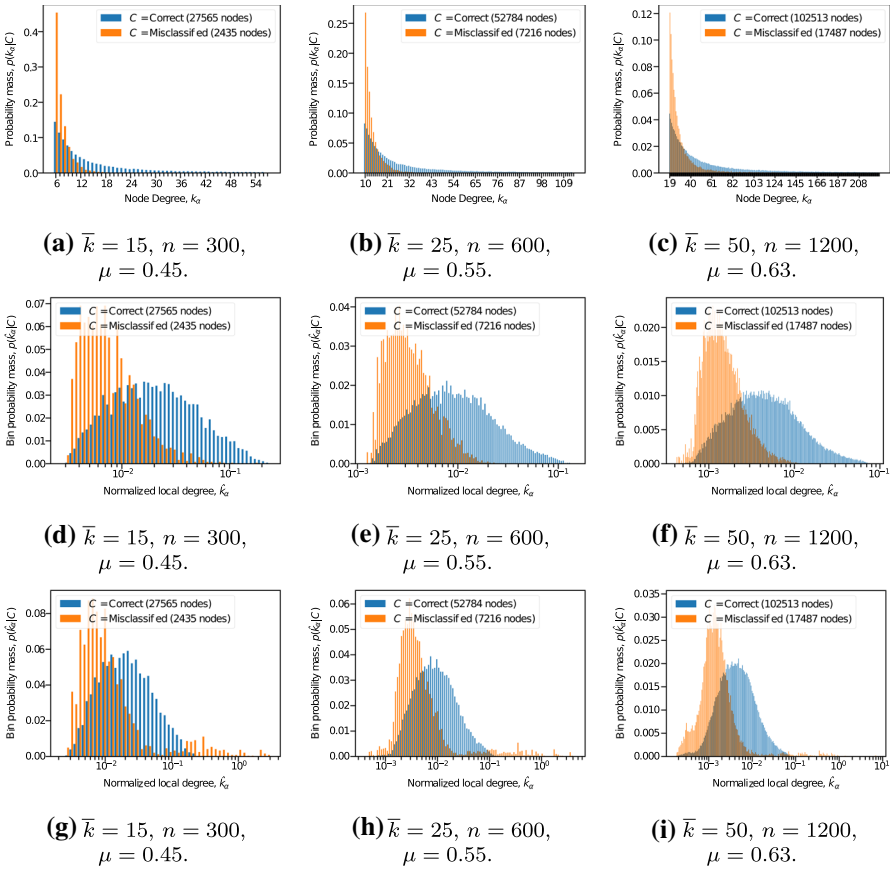


Fig. 2 Degree distributions for correctly classified and misclassified nodes in Synwalk results, obtained from 100 different LFR networks with common average degree, network size and mixing parameter. The top row shows the distributions of the node degrees, the middle row shows the distribution of the normalized local degrees w.r.t. the ground truth communities and the bottom row shows the distributions of the normalized local degrees w.r.t. the predicted communities. Synwalk tends to misclassify nodes with low normalized local degree (w.r.t. the ground truth communities), whereas the influence of the absolute node degree is negligible. The statistics of the normalized local degrees w.r.t. predicted communities indicate that misclassified nodes are assigned to additional, small communities (Color figure online)

amount of nodes in the sample network in Fig. 4, Synwalk achieves a significantly higher AMI value.

The above insights make apparent two advantages of our method. First, whereas there is no general answer on how to determine the random walk length T for Walktrap, Synwalk does not require the tuning of any hyper-parameter. Secondly, consider a network with many small communities and low average degree. Following our earlier observations, Walktrap will have many misclassified nodes due to the low average degree. In contrast, the assumption of small communities implies a reasonably high normalized local degree for the majority of nodes and thus suggests a better performance of Synwalk when compared to Walktrap. Indeed, the results in Fig. 5 support



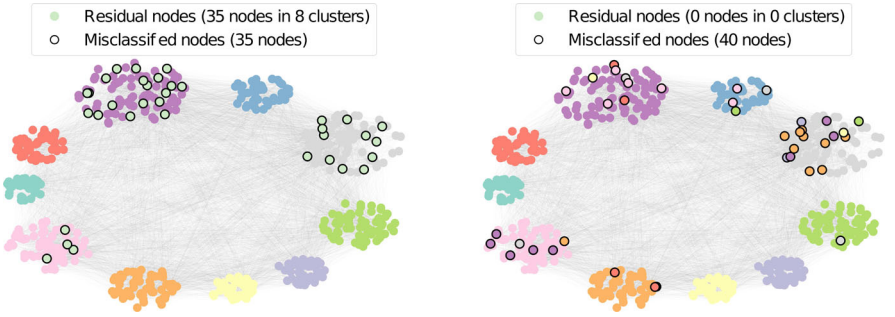
(a) $\overline{k} = 15$, $n = 300$, $\mu = 0.45$. **(b)** $\overline{k} = 25$, $n = 600$, $\mu = 0.55$. **(c)** $\overline{k} = 50$, $n = 1200$, $\mu = 0.63$.

(d) $\overline{k} = 15$, $n = 300$, $\mu = 0.45$. **(e)** $\overline{k} = 25$, $n = 600$, $\mu = 0.55$. **(f)** $\overline{k} = 50$, $n = 1200$, $\mu = 0.63$.

(g) $\overline{k} = 15$, $n = 300$, $\mu = 0.45$. **(h)** $\overline{k} = 25$, $n = 600$, $\mu = 0.55$. **(i)** $\overline{k} = 50$, $n = 1200$, $\mu = 0.63$.

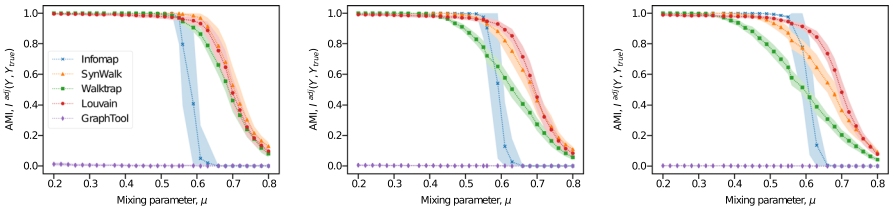
Fig. 3 Degree distributions for correctly classified and misclassified nodes in Walktrap results, obtained from 100 different LFR networks with common average degree, network size and mixing parameter. The top row shows the distributions of the node degrees, the middle row shows the distribution of the normalized local degrees w.r.t. the ground truth communities and the bottom row shows the distributions of the normalized local degrees w.r.t. the predicted communities. Walktrap tends to misclassify nodes with low normalized local degree (w.r.t. the ground truth communities) and/or low absolute node degree. The statistics of the normalized local degrees w.r.t. predicted communities resemble the ones w.r.t. the ground truth communities (Color figure online)

this intuition. The benchmark networks underlying these results were generated with parameter set B (see Table 1), effectively lowering the average community size for a given average degree compared to networks generated with parameter set A. Moreover, Synwalk closes the AMI performance gap to Louvain in this setup for sufficiently dense networks (cf. also similarly dense networks generated with parameter set A in Fig. 1).



(a) Synwalk, $I^{adj}(\mathcal{Y}, \mathcal{Y}_{true}) = 0.94$. **(b)** Walktrap, $I^{adj}(\mathcal{Y}, \mathcal{Y}_{true}) = 0.87$.

Fig. 4 A sample LFR graph with communities as detected by Synwalk and Walktrap. The network has $N = 600$ nodes, an average degree of $\overline{k} = 25$ and a mixing parameter of $\mu = 0.55$. Nodes are grouped according to their ground truth communities and share the same color if they belong to the same detected cluster. Misclassified nodes are highlighted with a black border. We aggregated nodes from predicted clusters that have no matching ground truth community into a single residual cluster. Synwalk places misclassified nodes into additional clusters, whereas Walktrap confuses node memberships between ground truth communities (Color figure online)



(a) $\overline{k} = 20$, $N = 600$, $\rho = 0.033$. **(b)** $\overline{k} = 20$, $N = 1200$, $\rho = 0.017$. **(c)** $\overline{k} = 20$, $N = 2400$, $\rho = 0.008$.

Fig. 5 Comparison of Infomap, Synwalk, Walktrap, Louvain and GraphTool on LFR benchmark networks with given average degree and network size. The lines and shaded areas show the mean and standard deviation of AMI as a function of the mixing parameter, obtained from 100 different network realizations. The networks were generated with parameter set B (see Table 1), simulating smaller communities with higher normalized local degrees. Synwalk outperforms Walktrap closes the performance gap to Louvain on sufficiently dense networks in this setup (Color figure online)

5.3 Illustration on empirical networks

In this section we illustrate the applicability of Synwalk on a selection of empirical networks (see Table 2) by comparing the detection results of Synwalk, Infomap, Walktrap, Louvain and GraphTool. For this purpose, we report the number of detected clusterings in Table 3, the number and fraction of non-trivial clusters thereof in Table 4, and the modularity score in Table 5.

Synwalk and Infomap behave similarly in terms of their single-number characteristics. An exception to this observation is the github network, where Synwalk detects a greater number of (non-trivial) clusters. Notably, Walktrap results consistently show a higher fraction of trivial clusters when compared to Synwalk and Infomap.

Table 2 Properties of the examined real-world networks

Network	Nodes	Links	$\bar{k}$	Source
dblp	317,080	1,049,866	6.62	(Yang and Leskovec 2015; Leskovec and Krevl 2014)
facebook	22,470	170,823	15.20	(Rozemberczki et al. 2019; Leskovec and Krevl 2014)
github	37,700	289,003	15.33	(Rozemberczki et al. 2019; Leskovec and Krevl 2014)
lastfm-asia	7624	27,806	7.29	(Rozemberczki and Sarkar 2020; Leskovec and Krevl 2014)
pennsylvania-roads	1,088,092	1,541,898	2.83	(Leskovec et al. 2008; Leskovec and Krevl 2014)
wordnet	146,005	656,999	9.00	(Fellbaum 1998; Kunegis 2013)

Table 3 Number of detected clusters for on the examined empirical networks

Network	Detected clusters				
	Infomap	Synwalk	Walktrap	Louvain	GraphTool
dblp	14,544	14,587	30,425	595	682
facebook	860	837	1227	62	156
github	1644	3935	7161	50	106
lastfm-asia	422	445	328	28	35
pennsylvania-roads	47,174	49,095	15,101	484	881
wordnet	5786	6058	14,063	516	541

The random walk-based methods Infomap, Synwalk, Walktrap yield a significantly higher amount of detected communities when compared to Louvain (modularity maximization) and GraphTool (SBM Inference)

The random walk-based methods Infomap, Synwalk, Walktrap yield a significantly higher number of detected communities when compared to Louvain (based on modularity maximization) and GraphTool (based on SBM Inference). GraphTool consistently detects almost no trivial clusters. Louvain detects no trivial clusters except on the pennsylvania-roads and wordnet networks. As expected, Louvain consistently yields the highest modularity scores.

Additionally, we compare the different methods by looking at the distribution of cluster sizes and normalized local degrees in their detected clusterings. For all distribution plots we consider clusters with less than three members as trivial and we do not include their statistics in the distributions. We provide the results for further cluster and node properties in Appendix E for the sake of completeness.

Infomap and Synwalk again behave similar given their cluster and node property distributions. A deviation from this pattern is apparent in the distribution of NLDs (see Fig. 6) for the github network, where Synwalk exhibits higher NLDs when compared to Infomap and Walktrap. The cluster size distributions (see Fig. 7) of Walktrap show a trend towards small clusters. As the empirical networks under consideration are significantly larger than the examined LFR networks in Sect. 5.2, a random walk length of $T = 4$ might not be the optimal hyperparameter choice and thus could explain the many trivial clusters detected (cp. Table 4).

Interestingly, whereas Synwalk achieved similar AMI performance as Walktrap in Sect. 5.2, Synwalk shows similar qualitative behavior to Infomap regarding cluster and node property statistics on empirical networks. However, the differences in the qualitative detection behavior of Synwalk and Walktrap that we discussed in Sect. 5.2.2 could explain the different results on the larger empirical networks. We further conjecture that the common search heuristic (see Sect. 5.2) of Synwalk and Infomap acts as a ”regularizer” on larger networks, i.e., the properties of predicted clusterings become more similar the larger the networks.

Last, we compare the detection results of Synwalk, Louvain and GraphTool w.r.t. the distributions of cluster sizes in Fig. 8 and normalized local degrees in Fig. 9. Cluster size distributions are compact and unimodal for Synwalk and GraphTool, whereas

Table 4 Number of non-trivial (less than three nodes) clusters on the examined empirical networks

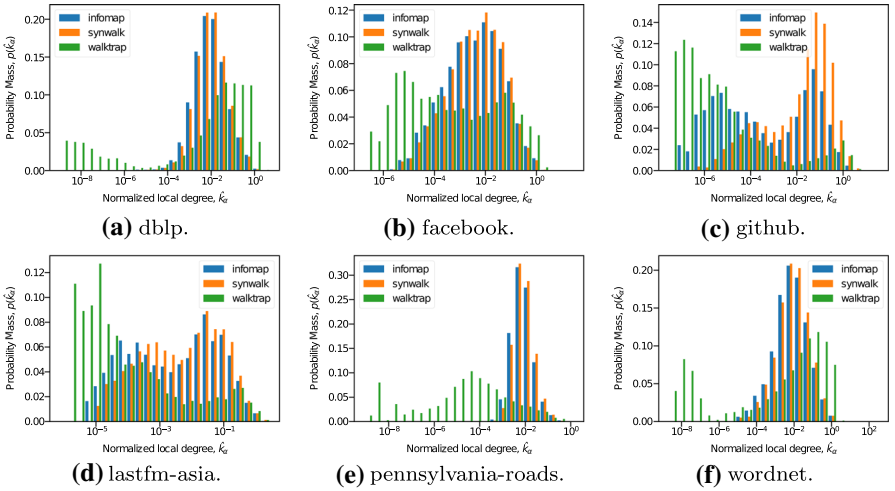
Network	Non-trivial clusters				
	Infomap	Synwalk	Walktrap	Louvain	GraphTool
dblp	14,533 (1.00)	14,576 (1.00)	25,659 (0.84)	595 (1.00)	682 (1.00)
facebook	813 (0.95)	807 (0.96)	834 (0.68)	62 (1.00)	155 (0.99)
github	1333 (0.81)	2228 (0.57)	630 (0.09)	50 (1.00)	104 (0.98)
lastfm-asia	374 (0.87)	411 (0.92)	202 (0.62)	28 (1.00)	35 (1.00)
pennsylvania-roads	47,004 (1.00)	48,925 (1.00)	14,931 (0.99)	314 (0.65)	881 (1.00)
wordnet	5604 (0.97)	5861 (0.97)	9531 (0.68)	350 (0.68)	540 (1.00)

We report the fraction of non-trivial clusters w.r.t. all detected clusters in brackets. Synwalk produces similar figures to Infomap, except for the github network. Walktrap predicts a higher fraction of trivial clusters compared to the other methods. GraphTool detects almost no trivial clusters. Louvain detects no trivial clusters except on wordnet and pennsylvania-roads

Table 5 Modularity scores on the examined empirical networks

Network	Modularity				
	Infomap	Synwalk	Walktrap	Louvain	GraphTool
dblp	0.73	0.73	0.67	0.81	0.72
facebook	0.76	0.74	0.75	0.80	0.62
github	0.39	0.27	0.36	0.45	0.14
lastfm-asia	0.74	0.70	0.77	0.81	0.68
pennsylvania-roads	0.90	0.89	0.95	0.99	0.95
wordnet	0.66	0.65	0.60	0.76	0.56

We list the number of detected clusters, the number of non-trivial (less than three nodes) clusters including the fraction thereof in brackets, and the modularity score for each clustering. As expected, Louvain (modularity maximization) achieves the highest modularity scores. The random walk-based methods Infomap, Synwalk, Walktrap produce similar scores across the networks. GraphTool (SBM Inference) tends to yiels lower scores when compared to Synwalk, except on the pennsylvania-roads network



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 6 Distributions of normalized local degrees w.r.t. the discovered communities on empirical networks for Infomap, Synwalk and Walktrap. The distributions generated by Synwalk resemble Infomap closely. An exception here is again the github network (Color figure online)

Synwalk yields consistently smaller communities than GraphTool by roughly and order of magnitude. Louvain cluster sizes vary in a broader range.

A consistent pattern is visible in the distributions of normalized local degrees: Synwalk delivers the highest and Louvain the lowest values/distribution centers with GraphTool in between. Remarkably, the distribution centers differ up to several orders of magnitude between methods. These observations indicate that Synwalk detects smaller communities with higher normalized local degree when compared to Louvain and GraphTool and supports our findings in Sect. 5.2.2.

Summarizing, in our comparison of the random walk-based methods (including Synwalk), Louvain and GraphTool their fundamentally different approaches to com-

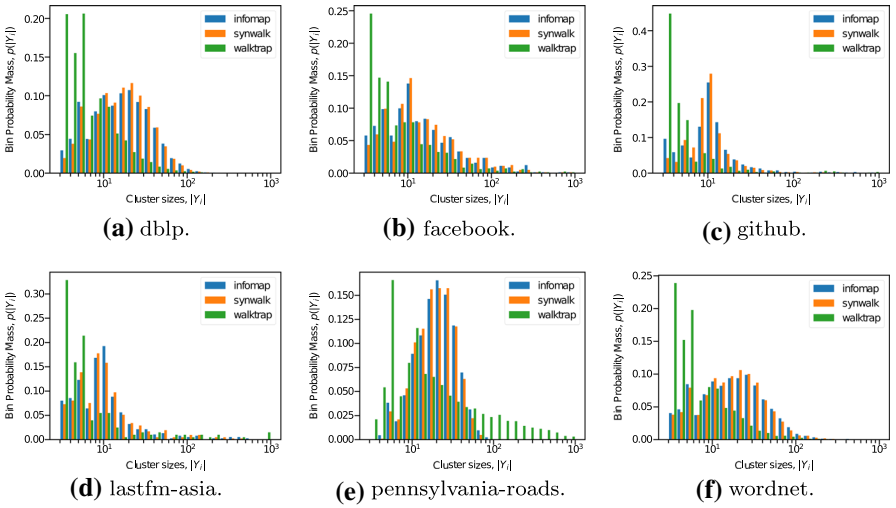


Fig. 7 Distributions of cluster sizes for the detection results on empirical networks for Infomap, Synwalk and Walktrap. Synwalk produces similar statistics to Infomap that are clearly distinguishable from Walktrap's results (Color figure online)

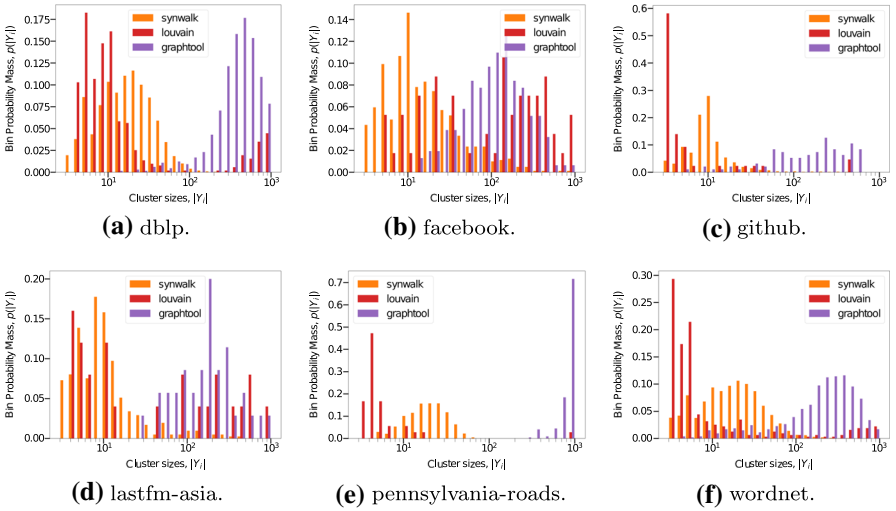
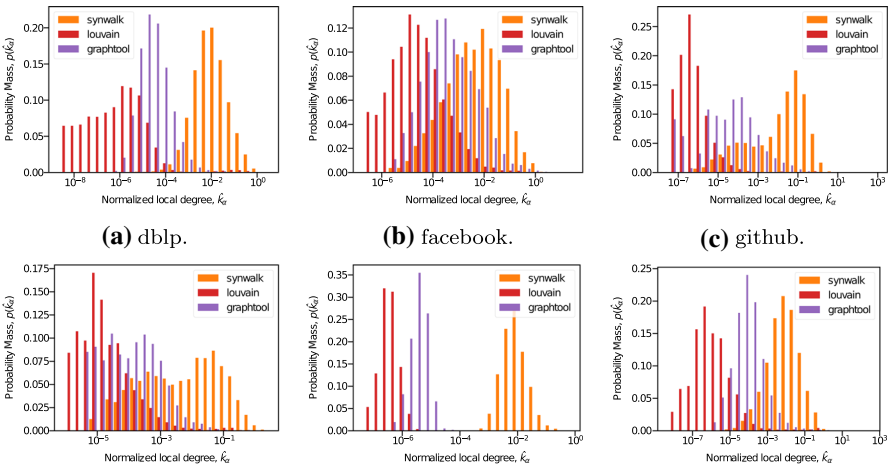


Fig. 8 Distributions of cluster sizes for the detection results on empirical networks for Synwalk, Louvain and GraphTool. Synwalk consistently detects smaller clusters thanGraphTool. Whereas cluster size distributions appear compact and unimodal for Synwalk and GraphTool, Louvain yields a wider spectrum of cluster sizes (Color figure online)

munity detection manifest in clear qualitative differences of their detected clusters. In light of the No Free Lunch theorem, each method captures different aspects of the networks' structure, each of which may be interesting to an expert analyzing networks in his domain of expertise.



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 9 Distributions of normalized local degrees w.r.t. the discovered communities on empirical networks for Synwalk, Louvain and GraphTool. Synwalk yields highest normalized local degrees, followed by GraphTool and Louvain, which delivers the lowest normalized local degrees. This observation is consistent across all networks. Interestingly, the distribution centers differ by up to several orders of magnitude (Color figure online)

6 Conclusion

In this work, we introduced Synwalk, a community detection method based on random walk modelling, that is characterized by an information-theoretic objective function. Our experiments underline the solid theoretical basis of synthetic random walk-based models and show that we can achieve robust performance across a wide range of problem setups. For specific networks, e.g., networks with many small communities and low average degree, Synwalk outperforms Infomap and Walktrap, at least on generated LFR benchmark graphs.

We deem random walk modelling an interesting counterpart to (stochastic) block modelling for community detection that deserves more attention, as it opens up many interesting avenues for future research. For example, our present study was limited to undirected networks only, suggesting a closer investigation of Synwalk in directed and/or weighted networks or networks with special properties (e.g., small worlds, etc.). Note that, as does Infomap, we estimate the transition probabilities and invariant distribution in (4) and (5) using the PageRank (Brin and Page 1998) algorithm with a non-zero teleportation probability (cp.Appendix B, Rosvall and Bergstrom 2008; Rosvall et al. 2009). This avoids the problem that a random walk may end up in an absorbing state in directed networks and naturally supports weighted networks. Hence, the Synwalk objective and our implementation are perfectly applicable to directed and/or weighted networks. Given the results in Sect. 5.3 we surmise that on directed empirical networks Synwalk will also yield qualitatively similar results to Infomap. Further, a deeper understanding of the optimization landscape induced by the Synwalk objective and the influence of the optimization algorithm is required,

as well as an extension of the approach to overlapping and hierarchical community structures.

Another interesting avenue for future work is to consider alternative definitions of the graph-induced random walks to the one we chose in (4), such as biased random walks, maximum entropy random walks, or even continuous-time random walks (cf. Masuda et al. 2017; Lambiotte et al. 2014). Recall that Synwalk tries to find a clustering that yields a synthetic random walk as close as possible to the network-induced random walk. One could view the synthetic random walk as a model of what we try to find out about the network, whereas the network-induced random walk as the lens through which we view this network. In that sense, choosing a certain lens will determine what aspects we can and cannot see about the network. Note that choosing a different network-induced random walk will not change the form of our objective function in (14). Only the transition probabilities and invariant distribution in (4) and (5) need to be computed accordingly. Thus, only little implementation effort would be necessary to realize these variants (cp. Appendix B). Yet, they may capture very different and interesting aspects of one and the same network.

Finally, as it is not only thinkable to change the network-induced random walk but also the synthetic random walk model, future research shall investigate random walk modelling approaches with a different synthetic random walk design from that in (8). As discussed above, the synthetic random walk is a model of what we are trying to find out about a network. Hence, the question arises whether such approaches can be tailored to detect communities of specific types or within specific network classes. Note that by changing the synthetic model necessarily the resulting objective function will be different from the one derived in this work, which in turn may entail new challenges in the optimization procedure.

Acknowledgements We thank Martin Rosvall and Christopher Blöcker for helpful comments during the early stages of this work.

Funding Open access funding provided by Graz University of Technology. The work of Bernhard C. Geiger was supported by the HiDALGO project and has been funded by the European Commission's ICT activity of the H2020 Programme under Grant Agreement Number 824115. The Know-Center is funded within the Austrian COMET Program - Competence Centers for Excellent Technologies - under the auspices of the Austrian Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology, the Austrian Federal Ministry for Digital and Economic Affairs, and the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

Availability of data and materials The generated benchmark graphs used to evaluate our method are available at Toth (2021).

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

Code availability The implementation of our community detection method is available at https://github.com/synwalk/synwalk. The source code of our evaluation framework is available at https://github.com/synwalk/synwalk-analysis.

Appendix A Proofs

A.1 Derivation of the Synwalk Objective

Proposition 1 *Let $\{X_t\}$ be a stationary Markov chain with transition probability matrix P derived from the weight matrix W of the network $\mathcal{G}$ and let $Q^{\mathcal{Y}}$ be a transition probability matrix of the same size parameterized as in (8). Then, for every partition $\mathcal{Y}$, it holds that*

$$\min_{\{[r^i_\alpha]_{\alpha\in\mathcal{Y}_i},\, s_i,\, u_i\}_{i\in\mathcal{K}^{\mathcal{Y}}}} \overline{D}(P\|Q^{\mathcal{Y}}) \le I(X_t; X_{t-1}) - \sum_{i\in\mathcal{K}^{\mathcal{Y}}} p_i D(p_{i\to i}\|p_i) \tag{18}$$

where p_i and $p_{i\to i}$ are defined as in (6).

Since $I(X_t; X_{t-1})$ is independent of the candidate partition $\mathcal{Y}$, minimizing the right-hand side of (18) over the partition $\mathcal{Y}$ is equivalent to a corresponding maximization of $\mathcal{J}(\mathcal{Y})$.

Proof Let $p_{\alpha\to i} := \sum_{\beta\in\mathcal{Y}_i} p_{\alpha\to\beta}$, $p_{\alpha\not\to i} := \sum_{j\neq i} p_{\alpha\to j} = 1 - p_{\alpha\to i}$, and let $\mathbb{I}_i(\alpha)$ denote the indicator function for cluster $\mathcal{Y}_i$, i.e.,

$$\mathbb{I}_i(\alpha) = \begin{cases} 1 & \text{if} \quad \alpha \in \mathcal{Y}_i \\ 0 & \text{otherwise.} \end{cases} \tag{19}$$

Then from (8) follows that

$$\begin{aligned}
\overline{D}(P\|Q^{\mathcal{Y}}) &= \sum_{\alpha,\beta} p_\alpha p_{\alpha\to\beta} \log \frac{p_{\alpha\to\beta}}{q_{\alpha\to\beta}} \\
&= \sum_{\alpha,\beta} p_\alpha p_{\alpha\to\beta} \left[\mathbb{I}_{m(\alpha)}(\beta) \log \frac{p_{\alpha\to\beta}}{r_\beta^{m(\beta)}(1-s_{m(\alpha)})} \right. \\
&\quad \left. +(1-\mathbb{I}_{m(\alpha)}(\beta)) \log \frac{p_{\alpha\to\beta}}{r_\beta^{m(\beta)} s_{m(\alpha)} \frac{u_{m(\beta)}}{1-u_{m(\alpha)}}} \right] \\
&= \sum_j \sum_\alpha \sum_{\beta\in\mathcal{Y}_j} p_\alpha p_{\alpha\to\beta} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} + \sum_i \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to i} \log \frac{p_{\alpha\to i}}{1-s_i}
\end{aligned}$$

$$
\begin{aligned}
&+ \sum_i \sum_{j\neq i} \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to j} \log \frac{p_{\alpha\to j}}{s_i \frac{u_j}{1-u_i}} \\
&= \sum_j \sum_\alpha \sum_{\beta\in\mathcal{Y}_j} p_\alpha p_{\alpha\to\beta} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} + \sum_i \sum_{j\neq i} \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to j} \log \frac{\frac{p_{\alpha\to j}}{p_{\alpha\not\to i}}}{\frac{u_j}{1-u_i}} \\
&\quad + \sum_i \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to i} \log \frac{p_{\alpha\to i}}{1-s_i} + \sum_i \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\not\to i} \log \frac{p_{\alpha\not\to i}}{s_i} \\
&= \sum_j \sum_\alpha \sum_{\beta\in\mathcal{Y}_j} p_\alpha p_{\alpha\to\beta} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} + \sum_i \sum_{j\neq i} \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to j} \log \frac{\frac{p_{\alpha\to j}}{p_{\alpha\not\to i}}}{\frac{u_j}{1-u_i}} \\
&\quad + \sum_i \sum_{\alpha\in\mathcal{Y}_i} \left[p_\alpha p_{\alpha\to i} \log \frac{p_\alpha p_{\alpha\to i}}{p_\alpha(1-s_i)} + p_\alpha p_{\alpha\not\to i} \log \frac{p_\alpha p_{\alpha\not\to i}}{p_\alpha s_i} \right] \\
&= \sum_j \sum_\alpha \sum_{\beta\in\mathcal{Y}_j} p_\alpha p_{\alpha\to\beta} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} + \sum_i \sum_{j\neq i} \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to j} \log \frac{\frac{p_{\alpha\to j}}{p_{\alpha\not\to i}}}{\frac{r_j}{1-r_i}} \\
&\quad + \sum_i D(p_\alpha p_{\alpha\to i} \| p_\alpha(1-s_i)). \qquad (20)
\end{aligned}
$$

We can now independently minimize the first summation term w.r.t $\{[r_\beta^j]_{\beta\in\mathcal{Y}_j}\}_{j\in\mathcal{K}^\mathcal{Y}}$, the second summation term w.r.t $[u_i]_{i\in\mathcal{K}^\mathcal{Y}}$, and the last summation term w.r.t $[s_i]_{i\in\mathcal{K}^\mathcal{Y}}$. Considering the latter, one can show (Cover and Thomas 2006, Lemma 10.8.1) that the Kullback–Leibler divergence $D(p_\alpha p_{\alpha\to i} \| p_\alpha(1-s_i))$ is minimized for

$$
1 - s_i = \sum_{\alpha\in\mathcal{Y}_i} p_\alpha p_{\alpha\to i} = p_{i\to i} \qquad (21)
$$

and hence

$$
s_i = 1 - p_{i\to i} = p_{i\not\to i}. \qquad (22)
$$

The minimizer regarding the distribution over nodes can be found along similar lines. For the first summation term in (20) we observe that

$$
\begin{aligned}
\sum_j \sum_\alpha \sum_{\beta\in\mathcal{Y}_j} p_\alpha p_{\alpha\to\beta} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} &= \sum_j \sum_\alpha p_\alpha p_{\alpha\to j} \sum_{\beta\in\mathcal{Y}_j} \frac{p_{\alpha\to\beta}}{p_{\alpha\to j}} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} \\
&= \sum_j p_j \sum_\alpha \frac{p_\alpha p_{\alpha\to j}}{p_j} \sum_{\beta\in\mathcal{Y}_j} \frac{p_{\alpha\to\beta}}{p_{\alpha\to j}} \log \frac{\frac{p_{\alpha\to\beta}}{p_{\alpha\to j}}}{r_\beta^j} \\
&= \sum_j p_j \sum_\alpha \frac{p_\alpha p_{\alpha\to j}}{p_j} \sum_{\beta\in\mathcal{Y}_j} \frac{p_{\alpha\to\beta}}{p_{\alpha\to j}} \log \frac{\frac{p_\alpha p_{\alpha\to\beta}}{p_j}}{\frac{p_\alpha p_{\alpha\to j}}{p_j} r_\beta^j}
\end{aligned}
$$

$$= \sum_j p_j \sum_\alpha \sum_{\beta \in \mathcal{Y}_j} \frac{p_\alpha p_{\alpha\to j}}{p_j} \frac{p_{\alpha\to\beta}}{p_{\alpha\to j}} \log \frac{\frac{p_\alpha p_{\alpha\to\beta}}{p_j}}{\frac{p_\alpha p_{\alpha\to j}}{p_j} r_\beta^j}$$
$$= \sum_j p_j D\left(\frac{p_\alpha p_{\alpha\to j}}{p_j} \cdot \frac{p_{\alpha\to\beta}}{p_{\alpha\to j}} \frac{p_\alpha p_{\alpha\to j}}{p_j} \cdot r_\beta^j \right)$$

and by again employing (Cover and Thomas 2006, Lemma 10.8.1), for a fixed cluster j this quantity is minimized for

$$r_\beta^j = \sum_\alpha \frac{p_\alpha p_{\alpha\to\beta}}{p_j} = \frac{p_\beta}{p_j} = \mathbb{P}(X_t = \beta | Y_t = j). \tag{23}$$

Applying these minimizers yields

$$\min_{\{[r_\alpha^i]_{\alpha\in\mathcal{Y}_i},\, s_i,\, u_i\}_{i\in\mathcal{K}^\mathcal{Y}}} \overline{D}(P\|Q^\mathcal{Y}) = \min_{[u_i]_{i\in\mathcal{K}^\mathcal{Y}}} I(X_t; X_{t-1}) - H(Y_t) + H(S_t|Y_t)$$
$$- \sum_i \sum_{j\neq i} p_i p_{i\to j} \log \frac{u_j}{1-u_i} \tag{24}$$

where $S_t = 1$ is a binary RV reflecting whether we stay in or leave a cluster at time t. Since for the distribution over clusters $[u_i]_{i\in\mathcal{K}^\mathcal{Y}}$ there exists no closed-form solution to the best of our knowledge, we choose $[u_i]_{i\in\mathcal{K}^\mathcal{Y}} = [p_i]_{i\in\mathcal{K}^\mathcal{Y}}$ as a sub-optimal solution. In other words, we utilize the stationary distribution of $\{Y_t\}$. Inserting this choice yields

$$\min_{\{[r_\alpha^i]_{\alpha\in\mathcal{Y}_i},\, s_i,\, u_i\}_{i\in\mathcal{K}^\mathcal{Y}}} \overline{D}(P\|Q^\mathcal{Y}) \leq I(X_t; X_{t-1}) - \sum_i p_i D(p_{i\to i}\|p_i). \tag{25}$$

This completes the proof. □

A.2 Bounds on the Synwalk Objective

Proposition 2 *Let $\{X_t\}$ be a stationary Markov chain with transition probability matrix P derived from the weight matrix W of the network $\mathcal{G}$ and let $\mathcal{Y}$ be any candidate partition. Then, we have*

$$0 \leq \sum_{i\in\mathcal{K}^\mathcal{Y}} p_i D(p_{i\to i}\|p_i) \leq I(Y_t; Y_{t-1}) \leq I(X_t; X_{t-1}) \tag{26}$$

where $\{Y_t\}$ is the process obtained by projecting $\{X_t\}$ through the partition $\mathcal{Y}$ (see Sect. 3).

Proof The first inequality follows immediately from the non-negativity of Kullback–Leibler divergence; the last inequality is the data processing inequality (Cover and

Thomas 2006, Th. 2.8.1), which follows from the fact that $Y_t - X_t - X_{t-1} - Y_{t-1}$ is a Markov tuple. We are thus left with proving the second inequality.

To this end, consider a relaxation of optimization problem (9), in which we let the distribution over clusters depend on the originating cluster; i.e., rather than a single distribution $[u_i]_{i\in\mathcal{K}^\mathcal{Y}}$, we now consider a set of distributions $\{[u_i^j]_{i\in\mathcal{K}^\mathcal{Y}}\}_{j\in\mathcal{K}^\mathcal{Y}}$. In other words, we consider, for every partition $\mathcal{Y}$, the minimization problem

$$\min_{\{[r_\alpha^i]_{\alpha\in\mathcal{Y}_i},\, s_i,\, [u_j^i]_{j\in\mathcal{K}^\mathcal{Y}}\}_{i\in\mathcal{K}^\mathcal{Y}}} \overline{D}(P\|Q^\mathcal{Y}). \tag{27}$$

It can be shown along the lines of Proposition 1 that the distributions $[r_\alpha^i]_{\alpha\in\mathcal{Y}_i}$ and s_i, for $i \in \mathcal{K}^\mathcal{Y}$, minimizing (9) also minimize (27). Thus, we have with (24) that

$$\begin{aligned}\min_{\{[r_\alpha^i]_{\alpha\in\mathcal{Y}_i},\, s_i,\, [u_j^i]_{j\in\mathcal{K}^\mathcal{Y}}\}_{i\in\mathcal{K}^\mathcal{Y}}} \overline{D}(P\|Q^\mathcal{Y}) = \min_{\{[u_j^i]_{j\in\mathcal{K}^\mathcal{Y}}\}_{i\in\mathcal{K}^\mathcal{Y}}} & I(X_t;X_{t-1}) - H(Y_t) + H(S_t|Y_t) \\ & - \sum_i \sum_{j\neq i} p_i\, p_{i\to j} \log \frac{u_j^i}{1-u_i^i}. \end{aligned} \tag{28}$$

The right-hand side can be shown to be minimized by setting

$$u_j^i = \frac{p_{i\to j}}{p_{i\not\to i}} = \mathbb{P}(Y_t = j|Y_{t-1} = i, Y_t \neq i) = \mathbb{P}(Y_t = j|Y_{t-1} = i, S_{t-1} = 1) \tag{29}$$

for $j \neq i$ and $u_i^i = 0$ for every $i \in \mathcal{K}^\mathcal{Y}$. Thus, we have

$$- \sum_i \sum_{j\neq i} p_i\, p_{i\to j} \log \frac{u_j^i}{1-u_i^i} \tag{30}$$

$$= - \sum_i \sum_{j\neq i} \mathbb{P}(Y_t = j, Y_{t-1} = i) \log \mathbb{P}(Y_t = j|Y_{t-1} = i, S_{t-1} = 1) \tag{31}$$

$$= - \sum_i \sum_{j} \mathbb{P}(Y_t = j, Y_{t-1} = i, S_{t-1} = 1) \log \mathbb{P}(Y_t = j|Y_{t-1} = i, S_{t-1} = 1) \tag{32}$$

$$= \mathbb{P}(S_{t-1} = 1) H(Y_t|Y_{t-1}, S_{t-1} = 1)$$

$$= H(Y_t|Y_{t-1}, S_{t-1}) \tag{33}$$

because $H(Y_t|Y_{t-1}, S_{t-1} = 0) = 0$. Inserting this into (28) yields

$$\min_{\{[r_\alpha^i]_{\alpha\in\mathcal{Y}_i},\, s_i,\, [u_j^i]_{j\in\mathcal{K}^\mathcal{Y}}\}_{i\in\mathcal{K}^\mathcal{Y}}} \overline{D}(P\|Q^\mathcal{Y}) \tag{34}$$

$$= I(X_t;X_{t-1}) - H(Y_t) + H(S_t|Y_t) + H(Y_t|Y_{t-1}, S_{t-1}) \tag{35}$$

$$\overset{(a)}{=} I(X_t;X_{t-1}) - H(Y_t) + H(S_{t-1}|Y_{t-1}) + H(Y_t|Y_{t-1}, S_{t-1}) \tag{36}$$

$$\stackrel{(b)}{=} I(X_t; X_{t-1}) - H(Y_t) + H(Y_t, S_{t-1}|Y_{t-1}) \tag{37}$$

$$\stackrel{(c)}{=} I(X_t; X_{t-1}) - H(Y_t) + H(Y_t|Y_{t-1})$$

$$\stackrel{(d)}{=} I(X_t; X_{t-1}) - I(Y_t; Y_{t-1}) \tag{38}$$

where (a) is due to stationarity, (b) due to the chain rule of entropy, (c) since S_{t-1} is a function of Y_t and Y_{t-1}, and (d) by the definition of mutual information. Since, finally, (27) optimizes over a larger feasible set (over $[u_i^j]_{i,j\in\mathcal{K}^\mathcal{Y}}$ rather than $[u_i]_{i\in\mathcal{K}^\mathcal{Y}}$), the minimum of (27) cannot exceed the minimum of (9). We thus have with (25) that

$$I(X_t; X_{t-1}) - I(Y_t; Y_{t-1}) \leq I(X_t; X_{t-1}) - \sum_i p_i D(p_{i\to i} \| p_i). \tag{39}$$

This completes the proof. □

A.3 Synwalk objective attains global optimum on network of isolated cliques

Proposition 3 *Let $\mathcal{G}$ be an unweighted network of disconnected cliques, defined by the partition $\mathcal{Y}^{\text{true}}$. Then, $\mathcal{Y}^{\text{true}}$ is a global optimizer of* (14).

Proof We prove Proposition 3 by first deriving the upper bound of the Synwalk objective for the given type of network, and then showing that the ground truth partition achieves this upper bound.

Let $\mathcal{G}$ be an unweighted network of disconnected cliques, i.e., there exists a partition function $\mathcal{Y}^{\text{true}}$ such that the link set E of $\mathcal{G}$ equals $\bigcup_{i\in\mathcal{K}^{\mathcal{Y}^{\text{true}}}} (\mathcal{Y}_i^{\text{true}})^2$. Note that self-loops are included in the link set. Since the network is unweighted, movement within cliques and the invariant distribution within each clique are uniform. Since the resulting Markov chain $\{X_t\}$ is not irreducible, infinitely many invariant distributions exist for the entire alphabet $\mathcal{X}$. Specifically, for every distribution $p^\bullet = [p_i^\bullet]_{i\in\mathcal{K}^{\mathcal{Y}^{\text{true}}}}$ over the ground truth communities, the distribution on $\mathcal{X}$ defined by $p_\alpha = p^\bullet_{m(\alpha)}/|\mathcal{Y}^{\text{true}}_{m(\alpha)}|$ is invariant under P. Thus, we have that

$$H(X_t) = -\sum_{\alpha\in\mathcal{X}} \frac{p^\bullet_{m(\alpha)}}{|\mathcal{Y}^{\text{true}}_{m(\alpha)}|} \log \frac{p^\bullet_{m(\alpha)}}{|\mathcal{Y}^{\text{true}}_{m(\alpha)}|} = -\sum_{i\in\mathcal{K}^{\mathcal{Y}^\bullet}} p_i^\bullet \log \frac{p_i^\bullet}{|\mathcal{Y}_i^{\text{true}}|}. \tag{40}$$

Given that the random walker is currently at node α in cluster $\mathcal{Y}_i^{\text{true}}$, all nodes in this cluster (including α) are equally likely to be visited in the next step. Thus, it follows that $H(X_t|X_{t-1} = \alpha) = \log|\mathcal{Y}^{\text{true}}_{m(\alpha)}|$. Combining this with $H(X_t)$ yields the upper bound

$$I(X_t; X_{t-1}) = H(X_t) - H(X_t|X_{t-1}) \tag{41}$$

$$= H(X_t) - \sum_{\alpha\in\mathcal{X}} p_\alpha H(X_t|X_{t-1} = \alpha) \tag{42}$$

$$= H(Y^\bullet), \tag{43}$$

where $Y^\bullet$ is a random variable with distribution $p^\bullet$.

Since the clusters coincide with the cliques in $\mathcal{G}$ there are no links connecting one cluster with another. Hence, we have $p_{i\to i} = 1$ for every cluster and thus $\mathcal{J}(\mathcal{Y}^{\text{true}}) = H(Y^\bullet) = I(X_t; X_{t-1})$, i.e., the upper bound from (15) is achieved. □

Note that a coarsening[2]
of the partition $\mathcal{Y}^{\text{true}}$ does not achieve this optimum. Indeed, while for $\widetilde{\mathcal{Y}}$ being a coarsening of $\mathcal{Y}^{\text{true}}$ it still holds that $\widetilde{p}_{i\to i} = 1$, we get

$$\widetilde{p}_i = \sum_{j:\mathcal{Y}_j^{\text{true}} \subseteq \widetilde{\mathcal{Y}}_i} p_j^\bullet \tag{44}$$

and thus $\mathcal{J}(\widetilde{\mathcal{Y}}) = H(\widetilde{Y}) < H(Y^\bullet)$ due to data processing (Cover and Thomas 2006, Problem 2.4).

Appendix B Implementation Details

As already noted in Sect. 4.1 for a given network (specifically, its weight matrix W) the Synwalk objective in (14) depends only on the candidate clustering $\mathcal{Y}$ given the transition probabilities $[p_{\alpha\to\beta}]_{\alpha,\beta\in\mathcal{X}}$ and stationary distribution $[p_\alpha]_{\alpha\in\mathcal{X}}$ (see Sect. 3.2). The transition probabilities can be computed according to (4). We estimate the stationary distribution using the PageRank (Brin and Page 1998) algorithm with a damping factor of 0.85 for any given network. Importantly, note that these quantities need only be computed once for each network and can be held fixed during the optimization procedure.

We tackle the combinatorial optimization problem over candidate clusterings by reusing Infomap's stochastic and recursive search algorithm as described in Rosvall and Bergstrom (2010, Appendix S1). To enable a fair comparison between Synwalk's and Infomap's objective functions we use the same set of default hyper-parameters for the search algorithm.

In a nutshell, the search algorithm splits into a core optimization phase and a tuning phase. Initially, each node serves as its own cluster. During the core optimization phase, neighboring clusters are greedily joined in a recursive fashion so as to maximize the given objective function. In the tuning phase there are two alternating mechanisms trying to further improve the objective. First, a fine tuning step moves single nodes between the clusters resulting from the core phase. Secondly, a coarse tuning step that divides the clusters resulting from the core phase into sub-clusters (by applying the core algorithm) and moves these sub-clusters between their parent clusters. For technical details we refer the reader to Rosvall and Bergstrom (2010, Appendix S1).

[2] We call a partition $\widetilde{\mathcal{Y}} = \{\widetilde{\mathcal{Y}}_1, \ldots, \widetilde{\mathcal{Y}}_M\}$ a *coarsening* of another partition $\mathcal{Y} = \{\mathcal{Y}_1, \ldots, \mathcal{Y}_K\}$ if each cluster in the coarsening $\widetilde{\mathcal{Y}}_i \in \widetilde{\mathcal{Y}}$ is comprised of one or more clusters of the original partition $\mathcal{Y}$ and $M < K$.

Appendix C Additional Results on the LFR benchmark

Here, we provide additional experiments on the LFR benchmark by investigating the AMI performance as a function of the network density.

In this experiment we again use parameter set A (see Table 1) to generate the LFR benchmark networks. We now fix the average degree and the mixing parameter of the generated LFR networks while varying their sizes in range $N \in [300, 19{,}200]$. We then plot the AMI as a function of the network density ρ in Fig. 10. All methods show improved AMI for increasing network densities, whereas Infomap's performance still mostly depends on the mixing parameter. Louvain consistently yields the hightest AMI values whereas GraphTool does not yield clusterings with relevant AMI values. Apparently, for Synwalk, Walktrap and Louvain there exist transition phases from low to high AMI with increasing network density. Interestingly, the location of these transition phases shifts to higher network densities the higher the absolute value of the average degree (Fig. 10, rows from left to right; cf. Sect. 5.2.1). In addition, the AMI results for Synwalk consistently rise above those of Walktrap during these transition phases. For very small network densities, Synwalk performance drops faster when compared to Walktrap. In summary, for a given average degree and mixing parameter, Synwalk performs better than or equal to Walktrap and Infomap, if the network size/density is sufficiently small/large.

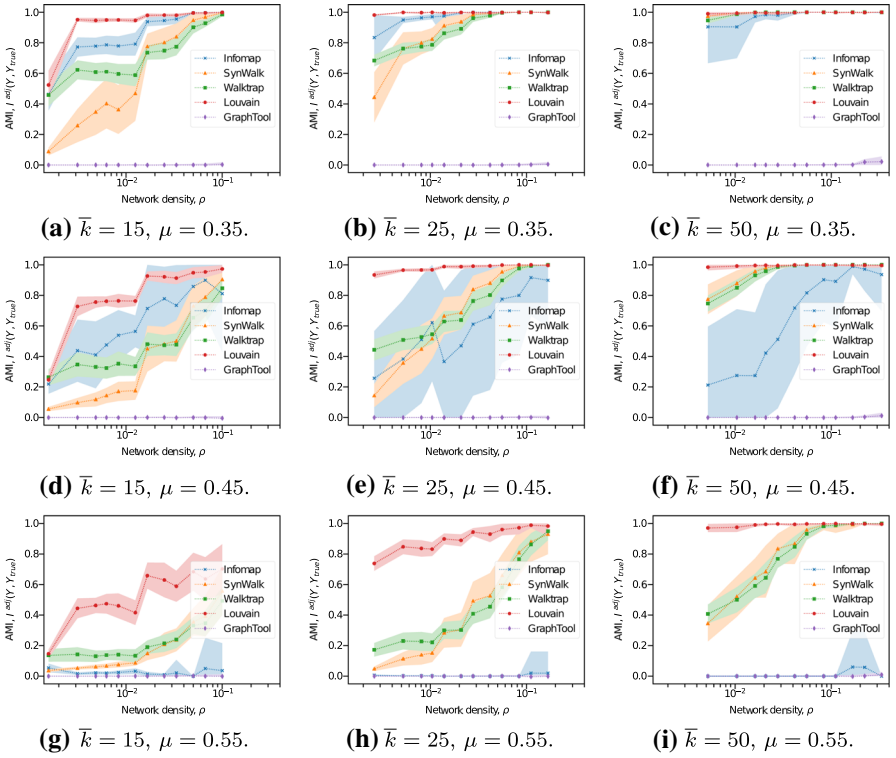
Appendix D Cluster Matching Algorithm

Consider the ground truth clustering $\mathcal{Y}^{\text{true}} = \{\mathcal{Y}_i | i \in \mathcal{K}^{\mathcal{Y}^{\text{true}}}\}$ and a prediction (by any community detection algorithm) $\mathcal{Y} = \{\mathcal{Y}_i | i \in \mathcal{K}^{\mathcal{Y}}\}$ with index sets $\mathcal{K}^{\mathcal{Y}^{\text{true}}}$ and $\mathcal{K}^{\mathcal{Y}}$ respectively. Then, in general, not only will the index sets differ in size but moreover they will encode equal or highly similar clusters in both clusterings differently. Hence, a matching between the index sets is necessary in our analyses (see Sect. 5.2.2).

For this purpose we employ the following greedy matching algorithm. The matching is based on the contingency table between the two clusterings $\mathcal{Y}^{\text{true}}$, $\mathcal{Y}$. In every step we add an entry to the cluster mapping by greedily picking the clusters with maximum overlap from the contingency table that do not already have a match. The procedure stops whenever all clusters of any input clustering have a match. The residual clusters obtain no mapping.

Appendix E Additional Results on the Empirical Networks

In this appendix we provide additional cluster and node statistics for the empirical networks considered in Sect. 5.3. The considered cluster properties are a subset



(a) $\overline{k} = 15,\ \mu = 0.35.$ **(b)** $\overline{k} = 25,\ \mu = 0.35.$ **(c)** $\overline{k} = 50,\ \mu = 0.35.$

(d) $\overline{k} = 15,\ \mu = 0.45.$ **(e)** $\overline{k} = 25,\ \mu = 0.45.$ **(f)** $\overline{k} = 50,\ \mu = 0.45.$

(g) $\overline{k} = 15,\ \mu = 0.55.$ **(h)** $\overline{k} = 25,\ \mu = 0.55.$ **(i)** $\overline{k} = 50,\ \mu = 0.55.$

Fig. 10 Comparison of Infomap, Synwalk, Walktrap, Louvain and GraphTool on LFR benchmark networks with given average degree and mixing parameter. The lines and shaded areas show the mean and standard deviation of AMI as a function of the network density (logarithmic scale), obtained from 50 different network realizations. Performance of Synwalk, Walktrap and Louvain rises with increasing network density, irrespective of the mixing parameter

of previously used measures for cluster characterization by Leskovec et al. (2010) and Yang and Leskovec (2015). Consider a cluster $S \subseteq \mathcal{X}$ within an undirected network $\mathcal{G} = (\mathcal{X}, E)$ and let

$$m_S = \frac{1}{2} \cdot |\{(\alpha, \beta) \in E \,|\, \alpha, \beta \in S\}| \tag{45}$$

denote the number of internal links of S, and

$$c_S = |\{(\alpha, \beta) \in E \,|\, \alpha \in S \wedge \beta \notin S\}| \tag{46}$$

denote the number of external links, i.e., links connecting nodes within S to nodes outside of S. Then we define the following cluster properties.

Algorithm 1: Greedy Cluster Matching Algorithm

Input: True clustering $\mathcal{Y}^{\text{true}}$, predicted clustering $\mathcal{Y}$
Output: Dictionary M: $\mathcal{K}^{\mathcal{Y}^{\text{true}}} \mapsto \mathcal{K}^{\mathcal{Y}}$
Initialize the dictionary $M \leftarrow \emptyset$
Initialize the number of true clusters $K^{true} = \max \mathcal{K}^{\mathcal{Y}^{\text{true}}}$
Initialize the number of predicted clusters $K^{pred} = \max \mathcal{K}^{\mathcal{Y}}$
Generate the contingency table $C \leftarrow$ `contingency_table(`$\mathcal{Y}^{\text{true}}, \mathcal{Y}$`)`
for $i \leftarrow 1$ **to** $\min\{K^{true}, K^{pred}\}$ **do**
  $k^{true} \leftarrow -1$
  $k^{pred} \leftarrow -1$
  while $k^{true} < 0$ **or** $k^{true} \in M$`.keys()` **or** $k^{pred} \in M$`.values()` **do**
    $k^{true}, k^{pred} \leftarrow \arg\max C$
    $C[k^{true}, k^{pred}] \leftarrow -1$
  end
  $M[k^{true}] \leftarrow k^{pred}$
end

– *Cluster density*. The cluster density

$$\rho(S) = \frac{m_S}{\binom{|S|}{2}} \tag{47}$$

is the density of internal links in S.
– *Clustering coefficient*. The clustering coefficient

$$c(S) = \frac{1}{|S|} \sum_{\alpha \in S} c(\alpha) \tag{48}$$

is the average of all node clustering coefficients $c(\alpha)$. Let $neigh(\alpha)$ denote the neighborhood of node α, i.e., all nodes that are connected to α by a link. Then the clustering coefficient for some node α is the fraction of realized triangles including α, i.e.,

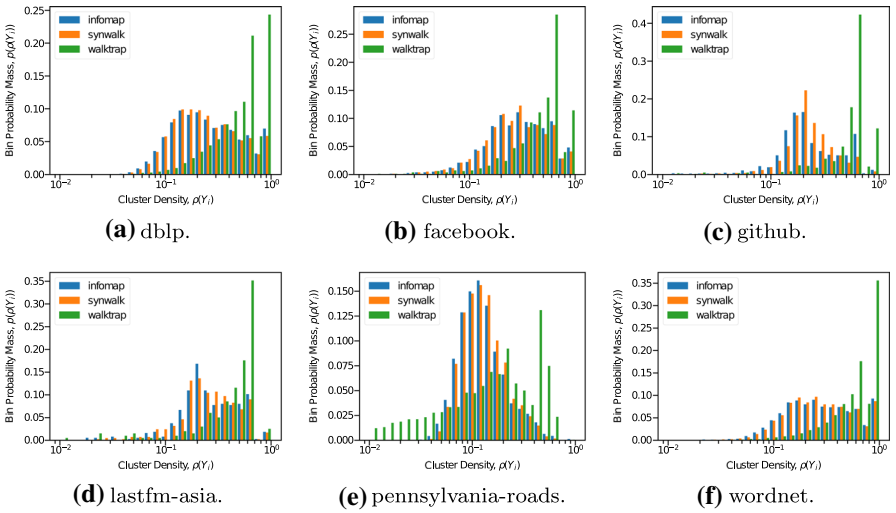
$$c(\alpha) = \frac{\frac{1}{2} \cdot |\{(\beta, \gamma) \in E \mid \beta, \gamma \in neigh(\alpha)\}|}{\binom{|neigh(\alpha)|}{2}}. \tag{49}$$

– *Conductance*. The conductance

$$\kappa(S) = \frac{c_S}{m_S + c_S} \tag{50}$$

gives the fraction of external links to the total number of cluster edges.
– *Cut ratio*. The cut ratio

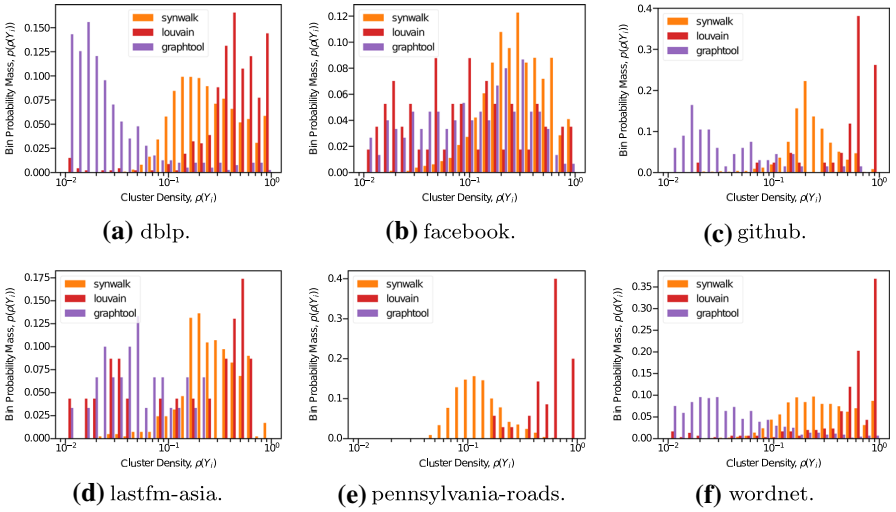
$$\xi(S) = \frac{c_S}{|S| \cdot (|\mathcal{X}| - |S|)} \tag{51}$$



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 11 Distributions of cluster densities for Infomap, Synwalk and Walktrap on empirical networks



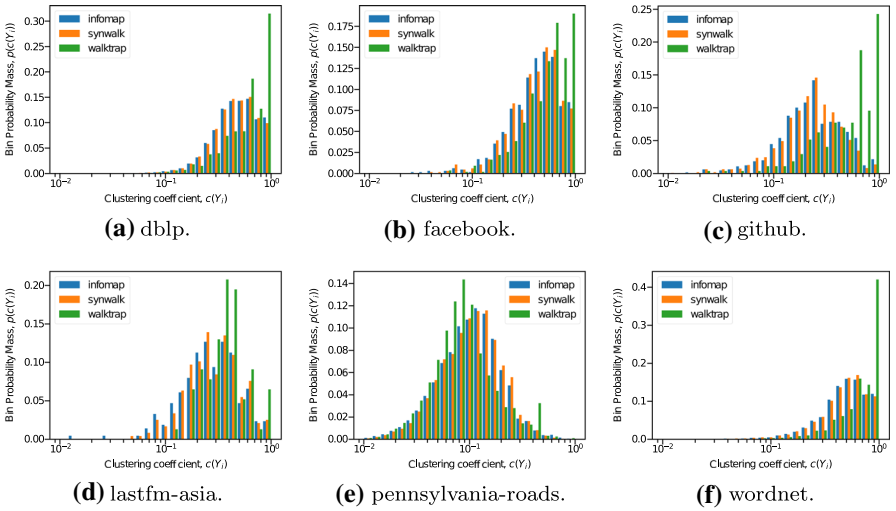
(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 12 Distributions of cluster densities for Synwalk, Louvain and GraphTool on empirical networks

is the ratio of external links to all possible external links.

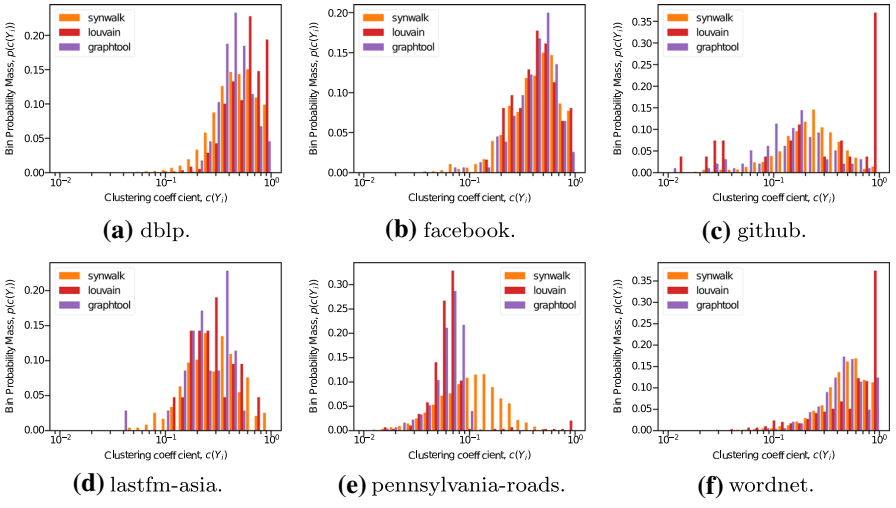
As already observed in Sect. 5.3, in general the cluster property statistics yielded by Synwalk and Infomap are highly similar, whereas Walktrap is mostly clearly distinguishable. The cluster densities (Fig. 11) are inverse proportionally distributed to the cluster sizes, e.g., Walktrap detects many small communities with high densities (Fig. 12). Clustering coefficients (Fig. 13) are higher for Walktrap on the github and wordnet networks when compared to Synwalk and Infomap (Fig. 14). Synwalk predicts highly conductive clusters for the github network (see conductance distributions in Fig. 15) (Fig. 16). Walktrap yields significantly smaller cut ratios (Fig. 17) com-



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 13 Distributions of cluster clustering coefficients for Infomap, Synwalk and Walktrap on empirical networks

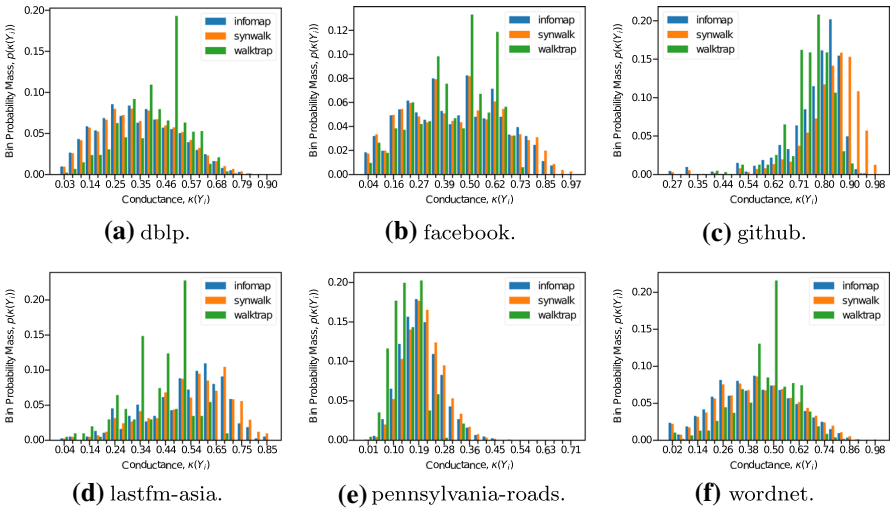


(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 14 Distributions of cluster clustering coefficients for Synwalk, Louvain and GraphTool on empirical networks

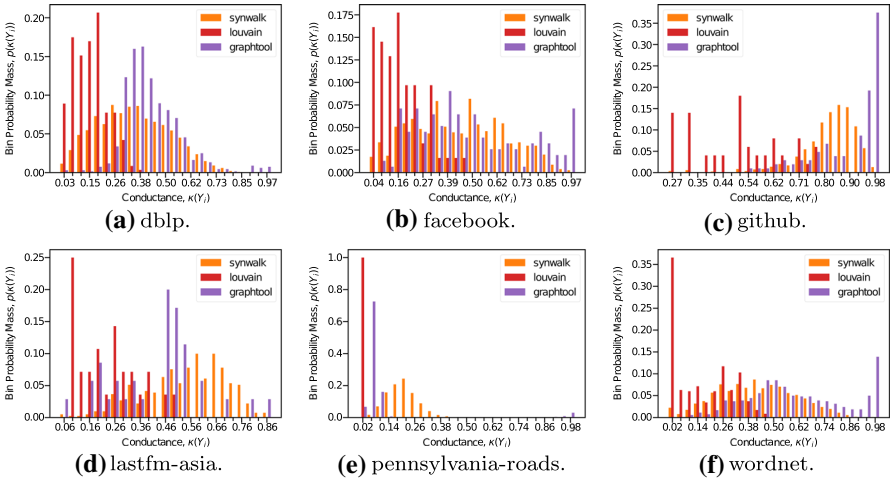
pared to Infomap and Synwalk on the lastfm-asia and pennsylvania-roads networks (Fig. 18). Distributions of the mixing parameters are displayed in Fig. 19 and show no notable difference between the three methods. An exception is the github network, where Synwalk predicts a clustering such that most of the nodes exhibit a mixing parameter $\gtrsim 0.5$. This reflects the high conductance values in Fig. 15.



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 15 Distributions of cluster conductances for Infomap, Synwalk and Walktrap on empirical networks

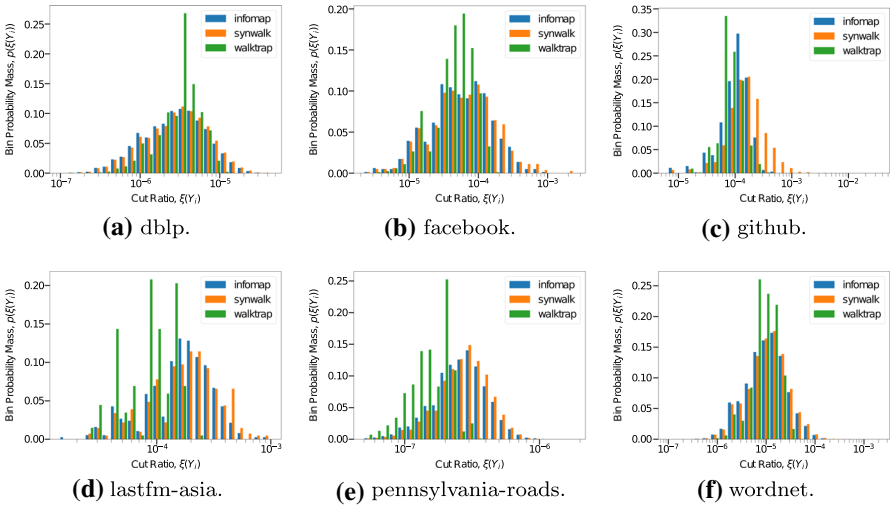


(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 16 Distributions of cluster conductances for Synwalk, Louvain and GraphTool on empirical networks

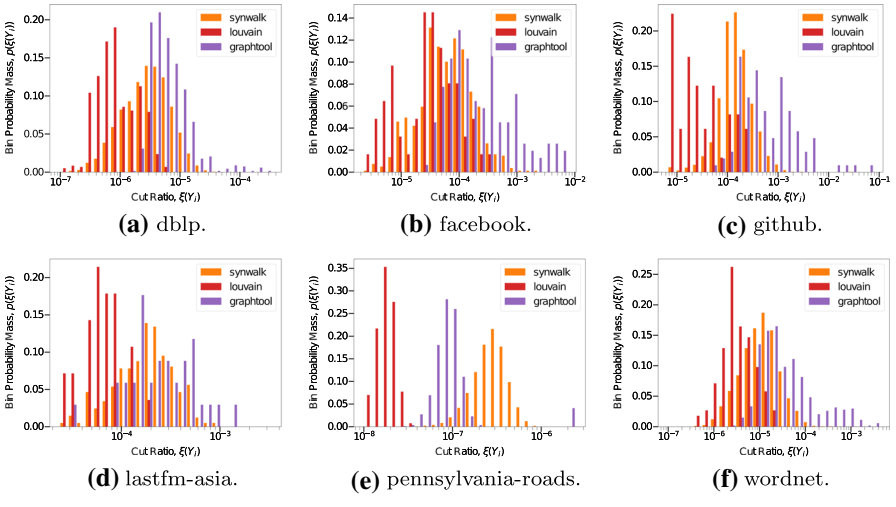
When comparing Synwalk to Louvain and GraphTool, cluster densities (Fig. 12) are lowest for GraphTool and highest for Louvain with Synwalk ranging in between. Clustering coefficients (Fig. 14) of Synwalk are significantly higher than those of Louvain and GraphTool only on pennsylvania-roads. Cluster conductances (Fig. 16) and cut ratios (Fig. 18) of Synwalk tend to lie in between those of Louvain (lower) and GraphTool (higher). The mixing parameters (Fig. 20) and show no grave difference between the methods.



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

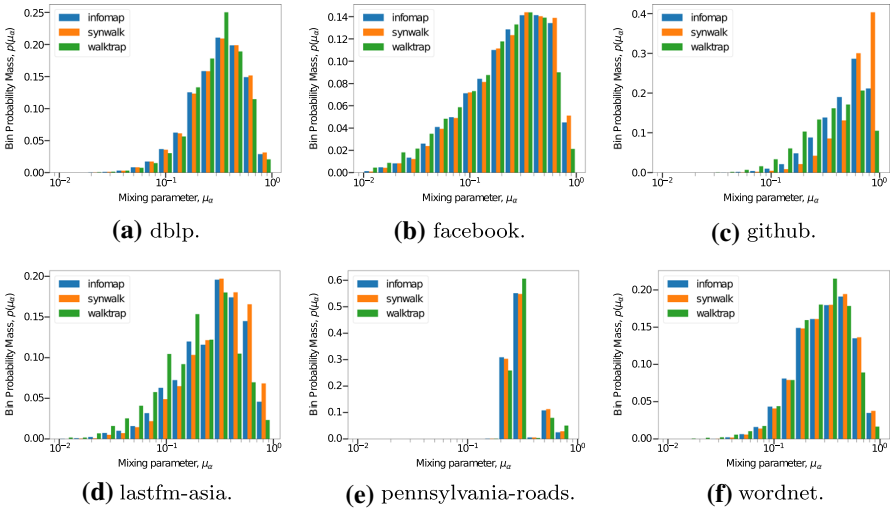
Fig. 17 Distributions of cluster cut ratios for Infomap, Synwalk and Walktrap on empirical networks



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

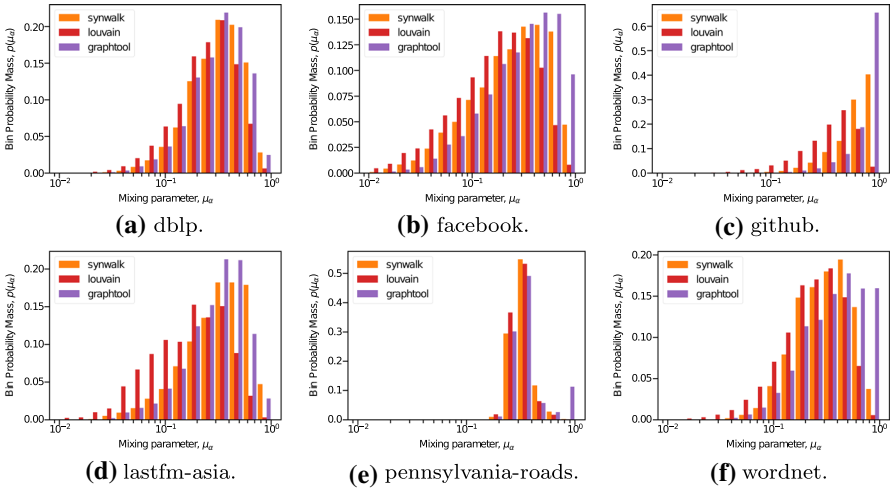
Fig. 18 Distributions of cluster cut ratios for Synwalk, Louvain and GraphTool on empirical networks



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 19 Distributions of node mixing parameters for Infomap, Synwalk and Walktrap on empirical networks



(a) dblp. **(b)** facebook. **(c)** github.

(d) lastfm-asia. **(e)** pennsylvania-roads. **(f)** wordnet.

Fig. 20 Distributions of node mixing parameters for Synwalk, Louvain and GraphTool on empirical networks

References

Amjad RA, Blochl C, Geiger BC (2020) A generalized framework for Kullback–Leibler Markov aggregation. IEEE Trans Autom Control 65(7):3068–3075. https://doi.org/10.1109/TAC.2019.2945891

Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 10:P10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Brandes U, Delling D, Gaertler M, Gorke R, Hoefer M, Nikoloski Z, Wagner D (2008) On modularity clustering. IEEE Trans Knowl Data Eng 20(2):172–188. https://doi.org/10.1109/TKDE.2007.190689

Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. In: Seventh International World-Wide Web Conference (WWW 1998)

Clauset A, Newman MEJ, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70(6):066111. https://doi.org/10.1103/PhysRevE.70.066111

Cover TM, Thomas JA (2006) Elements of information theory (Wiley series in telecommunications and signal processing), 2nd edn. Wiley-Interscience, New York

Csardi G, Nepusz T (2006) The igraph software package for complex network research. InterJournal Complex Syst 1695:1–9

Deng K, Mehta PG, Meyn SP (2011) Optimal Kullback–Leibler aggregation via spectral theory of Markov chains. IEEE Trans Autom Control 56(12):2793–2808. https://doi.org/10.1109/TAC.2011.2141350

Faccin M, Schaub MT, Delvenne JC (2020) State aggregations in Markov chains and block models of networks. arXiv:2005.00337 [physics.soc-ph]

Fellbaum C (ed) (1998) WordNet: an electronic lexical database. MIT Press, Cambridge

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174. https://doi.org/10.1016/j.physrep.2009.11.002

Fortunato S, Hric D (2016) Community detection in networks: a user guide. Phys Rep 659:1–44. https://doi.org/10.1016/j.physrep.2016.09.002

Ghasemi M, Hashemi A, Vikalo H, Topcu U (2020) Identifying sparse low-dimensional structures in Markov chains: a nonnegative matrix factorization approach. In: American Control Conf. (ACC), pp 1093–1098. https://doi.org/10.23919/ACC45564.2020.9147586

Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821–7826. https://doi.org/10.1073/pnas.122653799

Hurley N, Duriakova E (2015) Reformulations of the map equation for community finding and blockmodelling. In: Proc. IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM), Paris, France, pp 1606–1607. https://doi.org/10.1145/2808797.2809356

Hurley N, Duriakova E (2016) An information theoretic approach to generalised blockmodelling for the identification of meso-scale structure in networks. In: Proc. IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM), San Francisco, CA, pp 319–322. https://doi.org/10.1109/ASONAM.2016.7752252

Kesidis G, Walrand J (1993) Relative entropy between Markov transition rate matrices. IEEE Trans Inf Theory 39(3):1056–1057. https://doi.org/10.1109/18.256516

Kunegis J (2013) KONECT. In: Proc. Int. Conf. on World Wide Web—WWW '13 Companion, New York, NY, USA, pp 1343–1350. https://doi.org/10.1145/2487788.2488173

Lambiotte R, Delvenne JC, Barahona M (2014) Random walks, Markov processes and the multiscale modular organization of complex networks. IEEE Trans Netw Sci Eng 1(2):76–90. https://doi.org/10.1109/TNSE.2015.2391998

Lancichinetti A, Fortunato S (2009a) Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys Rev E 80(1):016118. https://doi.org/10.1103/PhysRevE.80.016118

Lancichinetti A, Fortunato S (2009b) Community detection algorithms: a comparative analysis. Phys Rev E 80(5):056117. https://doi.org/10.1103/PhysRevE.80.056117

Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110. https://doi.org/10.1103/PhysRevE.78.046110

Leskovec J, Krevl A (2014) SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data

Leskovec J, Lang KJ, Dasgupta A, Mahoney MW (2008) Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math 6(1):29–123

Leskovec J, Lang KJ, Mahoney M (2010) Empirical comparison of algorithms for network community detection. In: Proc. Int. Conf. on World Wide Web—WWW '10, New York, NY, USA, p 631. https://doi.org/10.1145/1772690.1772755

Masuda N, Porter MA, Lambiotte R (2017) Random walks and diffusion on networks. Phys Rep 716–717:1–58. https://doi.org/10.1016/j.physrep.2017.07.007

Newman MEJ (2006) Modularity and community structure in networks. Proc Natl Acad Sci 103(23):8577–8582. https://doi.org/10.1073/pnas.0601602103

Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69(2):026113. https://doi.org/10.1103/PhysRevE.69.026113

Orman GK, Labatut V (2009) A comparison of community detection algorithms on artificial networks. In: Gama J, Costa VS, Jorge AM, Brazdil PB (eds) Discovery science. Springer, Berlin, pp 242–256

Peel L, Larremore DB, Clauset A (2017) The ground truth about metadata and community detection in networks. Sci Adv 3(5):e1602548. https://doi.org/10.1126/sciadv.1602548

Peixoto TP (2014a) Efficient Monte Carlo and greedy heuristic for the inference of stochastic block models. Phys Rev E 89(1):012804. https://doi.org/10.1103/PhysRevE.89.012804

Peixoto TP (2014b) The graph-tool python library. figshare. https://doi.org/10.6084/m9.figshare.1164194

Peixoto TP, Rosvall M (2017) Modelling sequences and temporal networks with dynamic community structures. Nat Commun 8(582):1–12

Piccardi C (2011) Finding and testing network communities by lumped Markov chains. PLoS ONE 6(11):27028. https://doi.org/10.1371/journal.pone.0027028

Pons P, Latapy M (2005) In: Yolum P, Güngör T, Gürgen F, Özturan C (ed) Computing communities in large networks using random walks. Computer and Information Sciences, ISCIS, Istanbul, Turkey. https://doi.org/10.1007/11569596_31

Rached Z, Alajaji F, Campbell L (2004) The Kullback–Leibler divergence rate between Markov sources. IEEE Trans Inf Theory 50(5):917–921. https://doi.org/10.1109/TIT.2004.826687

Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D, Paris D (2004) Defining and identifying communities in networks. Proc Natl Acad Sci 101(9):2658–2663. https://doi.org/10.1073/pnas.0400054101

Raghavan UN, Albert R, Kumara S (2007) Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E 76:036106. https://doi.org/10.1103/PhysRevE.76.036106

Reichardt J, Bornholdt S (2006) Statistical mechanics of community detection. Phys Rev E 74:016110. https://doi.org/10.1103/PhysRevE.74.016110

Rosvall M, Bergstrom CT (2008) Maps of random walks on complex networks reveal community structure. Proc Natl Acad Sci 105(4):1118–1123. https://doi.org/10.1073/pnas.0706851105

Rosvall M, Bergstrom CT (2010) Mapping change in large networks. PLoS ONE 5(1):e8694. https://doi.org/10.1371/journal.pone.0008694

Rosvall M, Axelsson D, Bergstrom CT (2009) The map equation. Eur Phys J Spec Top 178(1):13–23. https://doi.org/10.1140/epjst/e2010-01179-1

Rosvall M, Delvenne JC, Schaub MT, Lambiotte R (2019) Different approaches to community detection. In: Doreian P, Batagelj V, Ferligoj A (eds) Advances in network clustering and blockmodeling. Wiley, Hoboken, pp 105–119. https://doi.org/10.1002/9781119483298.ch4

Rozemberczki B, Sarkar R (2020) Characteristic functions on graphs: birds of a feather, fromstatistical descriptors to parametric models. In: d'Aquin M, Dietze S, Hauff C, Curry E, Cudré-Mauroux P (eds) CIKM '20: the 29th ACM Int. Conf. on Information and Knowledge Management, Virtual Event, Ireland, pp 1325–1334. https://doi.org/10.1145/3340531.3411866

Rozemberczki B, Allen C, Sarkar R (2019) Multi-scale attributed node embedding. CoRR, arXiv:1909.13021

Toth C (2020) Synthesizing Infomap. Master's thesis, Graz University of Technology. https://doi.org/10.5281/zenodo.4446856

Toth C (2021) A collection of LFR benchmark graphs. https://doi.org/10.5281/zenodo.4450167

Vinh NX, Epps J, Bailey J (2010) Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. J Mach Learn Res 11:2837–2854

Yang J, Leskovec J (2015) Defining and evaluating network communities based on ground-truth. Knowl Inf Syst 42(1):181–213. https://doi.org/10.1007/s10115-013-0693-z

Yang Z, Algesheimer R, Tessone CJ (2016) A comparative analysis of community detection algorithms on artificial networks. Sci Rep 6(1):30750. https://doi.org/10.1038/srep30750

Zhang A, Wang M (2020) Spectral state compression of Markov processes. IEEE Trans Inf Theory 66(5):3202–3231