Journal of Complex Networks (2016) Page 1 of 22
doi:10.1093/comnet/cnw007

Detection of network communities with memory-biased random walk algorithms

Mesut Yucel
Department of Bioengineering, Faculty of Engineering, Ege University, Izmir, Turkey

Lev Muchnik
Department of Internet Studies, School of Business Administration, Hebrew University, Jerusalem, Israel

and

Uri Hershberg[†]
The School of Biomedical Engineering & Health Systems, Drexel University, Philadelphia, PA, USA
[†]Corresponding author. Email: uri.hershberg@drexel.edu

Edited by: Ernesto Estrada

[Received on 24 April 2015; accepted on 17 February 2016]

Community structure and its detection in complex networks has been the subject of many studies in the recent years. Towards this goal, we have created a novel approach based on the analysis of the motion of a memory-biased random walker, i.e. an entity that traverses the network with some tendency to follow or avoid pathways it has previously traversed. We found that the walker tends to remain inside communities, that is, subsets of the network nodes which are more connected to each other, rather than to the rest of the network. Based on this trait of the MBRW we developed a method to detect communities and tested its performance on a range of networks with different levels of community structure. In all tested cases, the method proved to be at least as effective as Girvan–Newman or Infomap while outperforming them when communities were less well defined.

Keywords: memory-biased random walker (MBRW); community detection; module density; cored networks.

1. Introduction

Networks are often used to describe the overall set of interactions in complex biological or social systems [1,2]. Such a holistic view of these systems is quite useful to understand system-wide processes, but it conflates the local interactions necessary for understanding of emergence of dynamic behaviour in societies, cells or multi-cellular organisms. To resolve its local interactions, we must ask how a network is segregated. Are there common rules of action in networks that allow segregation as an emergent phenomenon of action on the network? We suggest here a novel approach based on the stochastic movement of an entity on the network. The movement of an entity can be seen as a metaphor of information flow from a node to one of its neighbours and modelled as a sequence of local events governed by a memory of size s which biases by a factor α the likelihood of leaving a node by the remembered path. That memory capacity defines the timeframe within which the entity tends to retrace the paths it had recently travelled. In this work, we show that a random walk with a propensity to retrace its own recent

steps tends to converge to relatively dense network neighbourhoods and can be used to uncover the network communities [3,4].

Several groups have tried to develop algorithms to detect communities within a given network with respect to a chosen objective function as the quality measure of the extracted communities [3–9]. The most popular classes of community detection methodologies originate from the Girvan and Newman's divisive algorithm [3], spectral analysis [4,10,11] and flow-based approaches such as the map equation [12–15]. The first two of these classes are based on analysis of static networks. Although, in reality communities typically emerge from a continues dynamic process running on the network, topology-based algorithms rely on the information that comes from the structure of the network alone and not the dynamics on it. As a flow-based framework, the map equation could be considered to be capturing the modularity organization of a network from random movement. However, the existing techniques that use a map equation (e.g. Infomap's method [12]) rely on a stationary distribution of node visits of a random walker rather than actually simulating the walker's movement empirically. By doing so they lose valuable information about the transient state of the random walk movement. Our dynamical approach, on the other hand, deduces the spatio-temporal patterns of local interactions that constitute the network modules. Instead of relying on the information embedded in the static network or in the stationary distribution of the walkers' positions [12–15], our algorithm makes use of the trajectories of an entity continuously traversing the network along memory-biased paths.

Previous studies have shown that purely random graph surfing results in high correlation between nodes' degrees and the frequency of their being visited [16]. We show here that by adding a small bias towards the recently visited nodes, the entity's random motion starts to preferentially move on the core network elements with higher ratio of intra vs. inter-connectedness, and not simply on most connected nodes overall. Such nodes comprise the network's community-like structural elements and are often thought to be important features or functional modules in networks [12,13,17–22]. Here we show that the patterns of motion in a memory-driven random walk are indicative of the underlying network community structure. We designed a memory-biased random walker (MBRW), and created a random walk-biased community detection method that performs as well as, if not better than existing methods. The method particularly excels in detecting low-quality community elements, the kinds of network communities whose ratio of intra-links to inter-links is not as high as the intra-/inter-links ratio of well-defined theoretical communities one can find, for example, in Girvan–Newman networks [3]. This is an important result for uncovering the community structures of the real-world networks since such structures are mostly ambiguous in their community configuration [17].

2. Methods

2.1 *Analysing MBRW paths to identify communities*

Our community detection method identifies network communities using spectral analysis of their transition matrixes, weighted according to the patterns of motion generated by simulation of a MBRW. The analysis relies on identification of the recurrent sub-paths of the entity moving on networks. For instance, a path can be defined as a sequence of nodes visited by such randomly moving entity:

$$\text{Path} : [1, 5, 2, 1, 4, 7, 1, 5, 8, 1, 5, 8, 1, 5, 8, 12, 7, 1, 5]$$

The sequence of nodes contains repeating sub-path of size 3 (highlighted in the vector above). Such path can be used to compute the probabilities of relocation from a node (e.g. Node 1) to each of its neighbours (Nodes 3–5). The transition probabilities correspond to the transition frequencies inferred

from the recurrent sequences of steps. In the case above, the frequencies are:

$$w_{1,3} = 0, \quad w_{1,4} = 0, \quad w_{1,5} = 3$$

By definition, any two nodes belonging to the same community are expected to have more common neighbours than any two belonging to different communities. To reduce the probability of taking the inter-community link transitions into account, we thus counted the transitions between pairs of connected nodes only if they have at least 2 common nearest neighbours. Considering also the heterogeneous visiting distribution of the nodes, we constructed the transition matrix as $T = K^{-1}W_{i,j}$ where $K_{ii} = \sum_j W_{ij}$ and $W_{ij} = w_{ij}/\sum_l w_{il}$. Application of the spectral analysis to the normalized version of the transition matrix yields the community structure from the leading eigenvector resulting in the best clustering in terms of module density (as defined below) for the community number known a priori (or some other metric to find optimum community number). For clustering the eigenvector coefficients, we simply used the largest M_n gaps between the consecutive members of the sorted eigenvector coefficients to determine the cutting regions of the sorted coefficient sequence where M_n being the number of communities to be clustered.

2.2 *Simulating MBRW on artificial and real world networks*

We used a hybrid Gillespie algorithm, originally developed to describe the time evolution of chemical reaction systems [23,24], to simulate MBRW on the networks. At each step of this discrete movement, the entity occupying one of the network nodes moves to one of its immediate neighbours (Fig. 1). Given a memory capacity of size s, the entity faced with a choice of its destination among the few neighbours is inclined to revisit the nodes visited during the last s steps of the simulation. We prohibit backward movement, i.e. the return to the just visited node, to prevent repeating oscillation of the entity between a pair of connected nodes (in the Section 2.5.1 we explained how we deal with single-linked nodes which would be dead ends for the non-backtracking entity in this algorithm). Each simulation is performed with a fixed non-negative memory of size s and starts with the entity placed at a random node. Each step for the entity is chosen randomly based on a probability distribution vector inferred from the frequency of visits of the recent s steps of the simulation. We utilize the fitness proportionate (roulette-wheel) selection method [23–25] to infer the next step's destination μ. More specifically, the next destination of the entity is chosen among the current node's neighbours so that its index μ satisfies the following equation:

$$\sum_{v=1}^{\mu-1} a_v < r a_0 \leqslant \sum_{v=1}^{\mu} a_v$$

where r is a random number drawn from uniformly distributed unit interval and a_0 and a_v are the sum of all the elements of the probability distribution vector and vth element of it, respectively (note that the probability distribution vector as defined in our methodology is normalized so that $a_0 = 1$). Given a random number r, the next destination of the entity is determined using the above equation by iterating through the values of μ starting from $\mu = 1$ and increasing it until the equation is satisfied. We account for the memory by defining a parameter α that tunes the path bias towards the recently visited nodes. In particular, the element v of the vector of probabilities a is defined below for the exemplary vector of

$$a = [a_1, a_2, \ldots, a_s, \ldots, a_n]$$

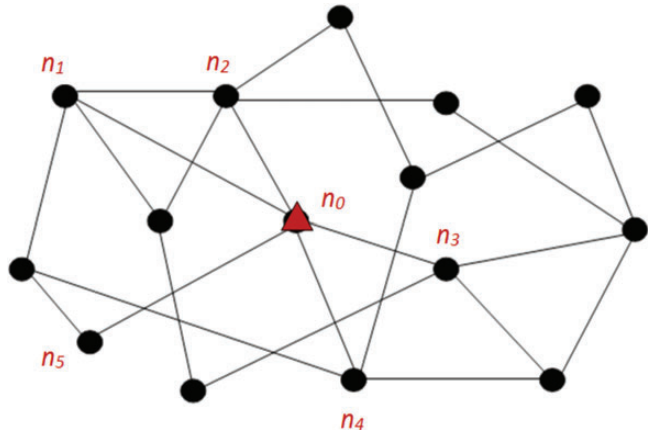

FIG. 1. Spatio-temporal updating of the entity's movement on a network according to a hybrid Gillespie algorithm. The entity, represented by a red triangle, is shown at the node n_0 having arrived from n_2 following the path $[n_4, n_0, n_1, n_2, n_0]$. The potential directions (next nodes) for the entity, if no back-tracking is allowed, can be given as a directional vector $D = [n_1, n_3, n_4, n_5]$. If there is no directional bias in the movement forward, then the approach is reduced to a random walk having a corresponding probability distribution vector of $a = [0.25, 0.25, 0.25, 0.25]$. On the other hand, if the entity has a memory $s = 5$, then the memory-biased probability distribution vector describing the probabilities of the next steps will be $a = [\alpha/(\alpha + 3), 1/(\alpha + 3), 1/(\alpha + 3), 1/(\alpha + 3)]$ where $\alpha > 1$ (since the entity tends to repeat its memories).

where n is the number of edges leaving the node, other than the edge arrived by (no backtracking) and a_s is the probability of taking the edge last used within memory to leave the node. Here, $a_s = \alpha/(\alpha + n - 1)$ while all other $a_{i \neq s} = 1/(\alpha + n - 1)$ (see Fig. 1 for example).

Note that movement with the memory size s set to 0, the entity will perform a pure random walk, selecting each direction with the same probability. However, even in this case, the perfect random walk scenario is altered by the inability of the entity to revert its last step, so that the future move would be a choice between all but one current node's peers. We exemplify the microscopic mechanism governing the entity movement in Fig. 1.

Termination criteria for the simulations. We measure the length of the simulation run necessary for convergence in terms of the *circulation number* C_n, the number of the simulation steps necessary for the moving entity to visit all the network nodes at least once. For example, the time it takes the walker to pay at least one visit to all the nodes is designated as $C_n = 1$. At that point, we reset the visitation counters and proceed with the simulation until all nodes are visited again ($C_n = 2$). To assure convergence, the simulation was executed until the results for C_n are identical for those obtained for $C_n + 1$.

2.3 *Analysing the movement sequence data*

To monitor convergence, we defined a property, that is, *module density*. Traditionally, people consider two general criteria that help quantifying the quality of a community detection: (1) the number of links among the members of the community as a measure of how coherent its nodes are and (2) the number of links between the members of the community and the rest of the network nodes as a measure of its relative isolation [26]. Module density is a multiplicative combination of these two criteria. First, we define relative connectedness (RC), a measure of intra-connectedness, as

$$\mathrm{RC} = \frac{2l_{in}}{n(n-1)}$$

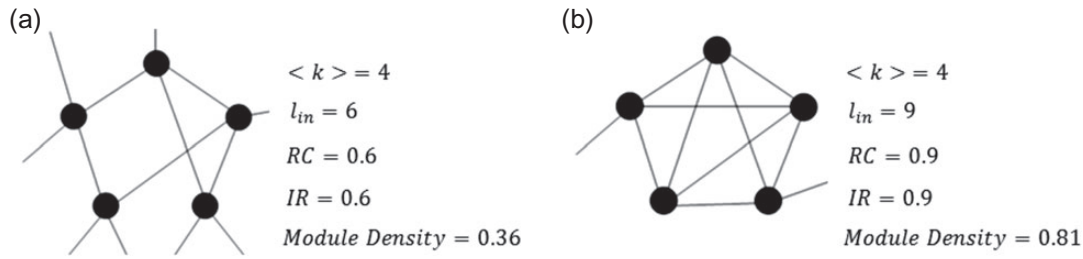

FIG. 2. Shows two sub-networks composed of five nodes each and having the same average connectivity $k = 4$. The links in the left sub-network (a) tend to be directed outward whereas the links in the right sub-network ($l_{in} = 6$) (b) point mainly inward ($l_{in} = 9$). Consequently, even though the remaining parameters do not differ between the sub-networks, the difference of RC, IR and module density between the sub-networks is significant.

where l_{in} is the number of ties actually linking the nodes of the candidate community and n is the number of nodes in the community. RC is in fact a density of links within the set of n nodes, ranging between 0 for a completely unconnected group of nodes and 1 for a fully connected node group (Fig. 2). The second measure, the intra-connection ratio (IR) is defined as

$$\mathrm{IR} = \frac{2l_{in}}{\sum_i^n k_i}$$

where k_i is the connectivity of the node i. IR represents the fraction of the existing links of the group of n nodes that are turned 'inward', link nodes in the group (as opposed to linking nodes outside the group). This number would be 1 if the group is completely isolated and 0 if none of the nodes is directly connected to any of the other nodes in the group (Fig. 2). Although RC and IR are both proportional to l_{in}, in the former it is a measure of the completeness of links of a node group whereas in the latter it is a measure of the inwardness of the group links. Their product (RC × IR) is thus in the range of 0–1 providing a convenient metric all together.

Using these two measures, we define *module density* as their product, that is, Module Density = RC × IR. Module density ranges between 0, representing a group of unconnected nodes (RC = 0 and IR = 0) which may or may not have links to the nodes outside the group, and 1 for a fully connected community (RC = 1) with no outgoing links (IR = 1) (Fig. 2). This metric differs from the Newman's modularity concept (a widely used quality index for communities [27]) although the two share a similar motivation. The Newman's modularity is based on the comparison between the fraction of the number of links between the community nodes and the number of expected links estimated from a random graph in which each node has the same degree as the original. Such random graphs can be generated using the well-known configuration model [28]. Due to the random model serving as a reference point, the modularity metric suffers from the resolution limit in resolving relatively small but well-defined communities [29]. For example, the expected number of links between two groups of nodes is typically diminishingly low for large sparse networks. Even a single accidental link between two separated community candidates may appear highly significant if the classical modularity measure is used. This may result in the decision to merge the communities as it would increase the overall network modularity. As opposed to the classical modularity, the module density introduced here, does not rely on a random null model, but uses an absolute metric for the community quality accounting for the local density and

potential links only. Contrary to the Newman's modularity, in the above example two small communities connected with a single link would be considered as two different communities by module density. Their RCs would be much higher than the RC of their merged community since merging the two groups would increase the potential intra-links dramatically. Although module density is an absolute metric, comparison of the community module density to that of the whole network would provide relative quality of partitioning. Therefore, one can use module density as the objective function to find optimum community numbers.

2.4 *Benchmark community identification methods*

In this work, we compared the performance of the MBRW with three widely accepted benchmark methods of community detection. We start by briefly describing these algorithms.

2.4.1 *Girvan–Newman's divisive algorithm.*

The Girvan–Newman algorithm [3] operates by gradually removing edges with the highest betweenness centrality until the network is split into the required number of connected components. Betweenness centrality of an edge is the fraction of the shortest paths between all node pairs that pass through that edge. Such edges tend to reside between dense network clusters so that they would participate in the paths running between densely connected clusters. Betweenness of such edges is high because shortest paths between the nodes belonging to different communities will run through these, rather rare, intercommunity edges. The Girvan–Newman algorithm exercises top-down hierarchical clustering by separating groups of nodes through gradual removal of the edges with highest betweenness centrality. Note that edge betweenness must be recomputed at each iteration because the value of betweenness depends on the entire network and changes with removal of each edge.

2.4.2 *Spectral analysis.*

Spectral analysis has been a broadly recognized alternative for the identification of network clusters. In the spectral network clustering, the eigenvectors of Laplacian L or transition matrix T or some modifications of them are used to extract the underlying community structure of a given network [4,10,11]. Transition matrix T of an adjacency matrix A is constructed as $T = AD^{-1}$ giving $\sum_j T_{ij} = 1$ with $T_{ij} \neq 0$ *iff* $A_{ij} = 1$ where D is the diagonal matrix whose element D_{ii} corresponds to the degree of node i and is given by $D_{ii} = \sum_j A_{ij}$ [6,30]. The matrix T contains the probabilities for a random walker to transition from node i to node j. If there is a bias in the probabilities of possible directions taken by the walker, one can use a generalized version of the normalized transition matrix, given as $T_{ij} = W_{ij} / \sum_l W_{il}$ where W_{ij} is the weighted matrix for directional preferences [6]. For instance, W_{ij} is taken as the similarity of nodes i and j in the topologically biased random walk (TBRW) [4,6] and is given by:

$$W_{i,j} = A_{i,j}(G_{i,j} + 1)$$

where $W_{i,j}$ is the topologically weighted matrix and $G_{i,j}$ is the number of common nearest neighbours of node i and j as a measure of their similarity. According to the Frobenius–Perron theorem, numerical proximity of the coefficients of the leading eigenvector corresponding to the second largest eigenvalue of T mirrors the proximity of the corresponding nodes in the network [4,10,30]. Based on this, one can cluster the network nodes by making use of an algorithm such as *k-means* clustering that partitions the

set of eigenvector coefficients into a given number of clusters where each coefficient (node) belongs to the cluster having the nearest mean [31].

2.4.3 *Infomap (MapEquation).* Infomap is a search algorithm operating in the map equation framework to identify modular organizations of networks with long flow persistence times [12]. Being a flow-based method, the map equation gives the theoretical lower limit of a modular description for the movement of a random walker in a network. It exploits the relationship between exploring community-like structures in networks and minimizing the description length of a random walker movement in those networks [12]. The information content for describing the pathways of a random walker in a network depends on the given number of communities to be explored in that network. A partition with the shortest description length will be the one that reveals the best set of communities in the network [12]. Rather than obtaining the random walker paths from empirical movement sequences, the map equation derives the description lengths from the stationary distribution of the random walker visits to the network nodes [12,14]. Infomap, whose core idea is very similar to the Louvain algorithm, minimizes the map equation over different partitions of networks [15]. In both approaches the main principle is to bring together neighbouring nodes into communities, lower-level communities into higher-level ones and so on. More specifically, Infomap starts by considering each node as a separate community. Then, it iteratively merges connected communities into larger ones if the merge results in the largest reduction in the map equation. This process continues until no move decreases the map equation further. Besides purely random walk, Infomap can also implement more complex mechanisms, such as higher-order Markov models of the walker movements. Such models can capture the effects of memory on the minimization of the description length [12]. As in the case of the random walk, the movement in the Markovian approaches too is characterized by the theoretical stationary distribution of the movement rather than by an empirical simulation of the movement.

We used the MapEquation web platform to conduct the Infomap community analysis of the test networks [32].

2.5 *Test networks*

We tested the MBRW algorithm on several types of constructed networks, including the Girvan–Newman network model [3], our own model of cored networks, a set of benchmark models designed for the evaluation of community detection algorithms [17] and on a real-world protein–protein interaction network from *Caenorhabditis elegans* [22]. The following section describes these networks in details.

2.5.1 *Girvan–Newman networks.* Girvan–Newman network model was originally designed to generate random networks with community structure and are commonly used to validate community detection algorithms [3]. The network community structure is defined by the inter- and intra-community connectivity. More specifically, the number of inter-community links per node, k_{int}, defines the inter-community connection probability, p_{out}, as $p_{\text{out}} = k_{\text{int}}/2(N - n)$ where N is the total number of nodes in the network and n is the number of nodes in the communities. Thus, p_{out} is the probability for a node to be connected with any other node in a different community. Together, p_{out}, k and n define the intra-community connection probability as $p_{\text{in}} = (k - k_{\text{int}})/2(n - 1)$. Similar to p_{out}, p_{in} expresses the probability for a node to be connected with each node residing in the same community. Iteratively, the algorithm connects pairs of nodes belonging to the same community with a probability of p_{in} and pairs

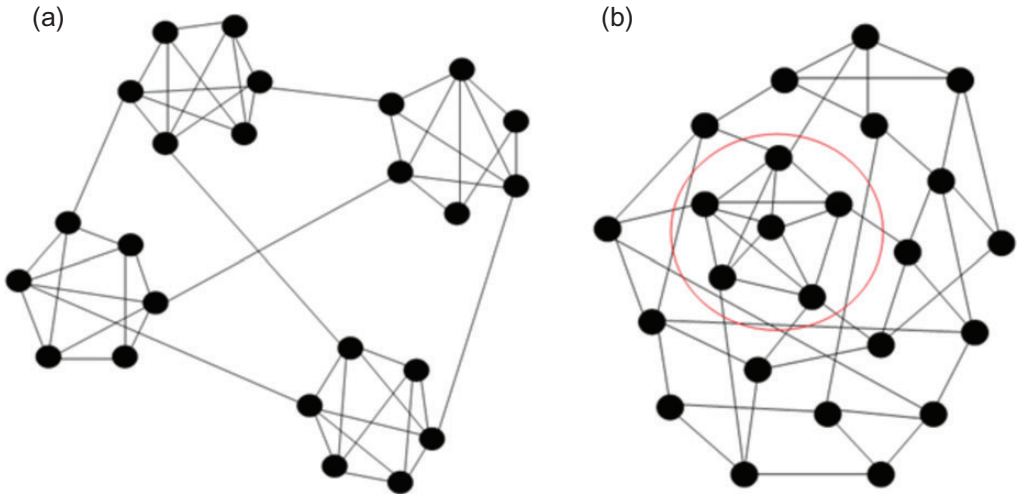

FIG. 3. Illustration of the Girvan–Newman networks with 24 nodes and 4 equal-sized communities (a) and of cored networks with 24 nodes and a core of size 6 (circled in red) (b).

belonging to different communities with the probability p_{out}. We generated a set of Girvan–Newman networks of $N = 128$ nodes assigned to four communities of $n = 32$ nodes, with average degree of $k = 16$ and varying number of inter-community links k_{int} (Fig. 3(a)). The number of inter-community links ranged from $\beta = 1.5$ to $k_{\text{int}} = 8$, at which point a node had as many inter-links as intra-links. In addition, we warranted (in Girvan–Newman and all other network models in this work) that all nodes had a degree of at least 2. This prevented the moving entities 'short circuiting' the network and getting stuck in dead ends they could not return from. To avoid such situations, we connected nodes with degree $k < 2$ to $2 - k$ random nodes.

2.5.2 *Cored networks.* Design of the Girvan–Newman networks is such that the communities are separated from one another with relatively clear boundaries so that every part of the network represents a different community and all parts of the network belong to a community. We find an even division of the entire network into a set of similarly structured communities to be somewhat artificial. Many real-world networks tend to have communities of different sizes overlap and even regions that do not belong to any clear community. We therefore created another set of artificial networks designated as *cored networks* (Fig. 3(b)). Cored networks are random networks with embedded densely connected regions (cores) of different sizes. Thus, instead of being a collection of communities as in Girvan-Newman networks, cored networks are composed of a complex hierarchical structure of smaller communities nested within larger ones. Construction of the cored networks starts with creation of the cores. The n core nodes are randomly interconnected with an average core connectivity of k_{core}. The core is next embedded in the network by adding links between the core nodes and the background nodes until the average inter-connectivity (k_{int}) is reached. More specifically, the core node i, ($i = 1, \ldots, n$) connects to a background node j, ($j = 1, \ldots, N - n$) with a probability of

$$p_{\text{out}} = \frac{k_{\text{int}}}{N - n}$$

The remaining links connect background nodes so as to achieve the average overall network connectivity of k. A pair of background (non-core) nodes is thus connected with a probability of

$$p_{\text{non-core}} = \frac{Nk - nk_{\text{core}} - 2nk_{\text{int}}}{(N-n)(N-n-1)}$$

We used cored networks containing $N = 100$ nodes and composed of a core of $n = 20$ nodes. They had an average connectivity of $k = 10$ and k_{int} ranging from 1 to 5, for which a core node had as many inter-connections as intra-connections, a scenario equivalent to the Girvan–Newman networks above but with a more complex, hierarchical and less uniform division of communities.

2.5.3 *Non-homogeneous community benchmark networks.* To test our algorithm on larger non-uniform networks we used the more novel benchmark networks described in [17]. The artificial network can control the non-homogeneity in node connectivity, community size and community quality distributions which more reliably model the range of structures found in real-world networks. We created these network such that connectivity and community size distributions were heterogeneous obeying power laws with degree exponents γ and β, respectively [17]. To model larger networks of sizes relevant in their structure and size to real-world biological and social networks, we generated random networks with the number of nodes $N = 5000$, mean degree $k = 20$ and power law degree and cluster size distributions of $\gamma = 2.5$ and $\beta = 1.5$. The*mixing parameter* μ received the values $\mu = 0.1$, 0.2, 0.3, 0.4 and 0.5 to control the level of ambiguity of the created communities in the network by setting the fraction of node's out-links pointing outside of its community.

We tested the speed of our algorithm by analysing its performance on an ensemble of larger networks. For this, we created 10 networks for each value of $N = 1 \times 10^4$, 2×10^4, …, 10×10^4 with average degree $k = 20$, mixing parameter $\mu = 0.2$ and community sizes between $0.004N$ and $0.04N$ for each N.

2.5.4 *Protein–protein interaction network of C. elegans.* To test the capability of our approach to pinpoint community-like structures in real-world networks we studied the protein interaction network of *C. elegans* [22]. Protein interaction networks represent interaction data of proteins, where each protein is represented by a node and links represent interaction partners of that protein [22]. For our analysis, we extracted the giant component of the protein interaction network having 197 proteins and 464 interactions. As in the other cases we prevent entities from getting stuck by eliminating the nodes with fewer than two connections. Node removal does not impact the overall structural organization of the network because the nodes with a degree of 1 are peripheral. Note that this procedure is equivalent to extraction of 2-*core* with *k-core* decomposition [33] as the node removal process is iterative and continues until no nodes with a degree lower than 2 remains (removal of all nodes with a degree of 1 may lead to emergence of new nodes with that qualify for removal).

After removal of nodes with only a single edge the protein interaction network had 128 nodes and 382 links with an average connectivity of $k = 5.97$ (as opposed to 4.71 before the elimination of dead end nodes).

2.6 *MBRW simulation parameters*

In MBRW an entity moves on the networks preferentially. The entity's destination from a node is determined by the last s steps of its movement (a memory of size s) and the 'strength' of its tendency

(α value) to prefer a destination/s which has already been visited within these last s steps over the other potential destinations. We experimented with a range of simulation parameters (s and α) and observed that the results were not overly sensitive to the choice of their values so long as they were $s \geqslant 5$ and $\alpha \geqslant 1000$. Considering a parameter space of $s = [4, 5, 10, 15, 25]$, $\alpha = [10, 100, 1000, 5000, 10\,000]$ and $C_n = [1, 2, \ldots, 10]$ we found (data not shown) that when a larger memory size ($s \geqslant 10$) was employed, the memorized long paths mostly included a subsection of repeating small paths of size 3–4 steps. Thus, the capacity of longer memory effectively reduced to the capacity of a much smaller one with greater cost in simulation step number needed for the entity to visit all the nodes at least. We also observed that although any value of $\alpha > 1$ improved, in general, the results of otherwise pure random walk, $\alpha = 1000$ was sufficient for most of the networks to obtain the optimal results. Based on this finding, in this study we simulated the movement of the entity with a memory of $s = 5$ (and without memory, $s = 0$ for comparison). When $s = 5$ we used $\alpha = 1000$ implying that a memorized direction will be taken 1000 times more probably than an un-memorized one (Fig. 1).

For every condition of artificial networks, we created 10 networks and ran 10 simulations of entity movement for each of these networks. These included: (i) 8 Girvan–Newman networks with $k_{\text{int}} = 1$ to 8, (ii) 5 cored networks with $k_{\text{core}} = 10$ and $k_{\text{int}} = 1$ to 5, (iii) 5 benchmark networks [17] with $N = 5000$ and $\mu = 0.1$ to 0.5 and (iv) 10 benchmark networks [17] with $\mu = 0.2$ and N ranging from 10^4 to 10^5 at $10k$ node steps. Lastly we ran 25 simulations of movement on the *C. elegans* protein interaction network system. The circulation number was not constant but dependent on the k_{int} value of the networks, i.e. for bigger k_{int}-valued networks the results (mean community qualities or module densities) converged to their steady state values in larger circulation numbers. In other words, when networks were more interconnected or the communities less delineated it took the entity longer to identify them through movement.

Similar to the *k-means analysis* [31], the largest coefficient gap analysis of the leading eigenvector of the MBRW-weighted transition matrix requires the partition number to be defined *a priori* (see the Section 2.4.2). In the case of the *C. elegans* protein interaction network, we needed to make an estimation for the community number since it is an un-designed (real-world) network with unknown number of communities. To do so we employed largest difference analysis of the eigenvector coefficients in which we used the module density to assess the comparative community qualities of the candidate community sets. We clustered the leading eigenvector by cutting its coefficient sequence at $M_n - 1$ positions where the consecutive eigenvector coefficient pair has the largest linear distance. We applied this procedure for different partition numbers from $M_n = 1$ to 30. To eliminate the dependence of the results on the stochastic nature of the underlying algorithm, we applied this analysis to each of the 25 instances of a simulated network with given partition number and used the mean module density for each partition number.

2.7 *Performance analysis of MBRW*

We assess the algorithmic performance of MBRW empirically, by running it on a set of large networks. The benchmark networks we used were generated as defined in [17] with $N = 1 \times 10^4$, $2 \times 10^4, \ldots, 10 \times 10^4$ nodes. We executed 10 simulations with a circulation number of 1 on 10 instances for each of the network sizes. We measured the elapsed (physical) times for each circulation and presented them with the simulation step number (the size of the movement sequence) with respect to the network size. The computations were performed on a workstation equipped with a single Intel (R) Xeon CPU W3680 3.33 GHz. We employed a linear regression model to construct a relationship between the elapsed physical time per network node and the network size.

TABLE 1 *Percentage of correctly classified nodes for the Girvan–Newman networks with different inter-community link numbers per node* (k_{int}). k_{com}, C_n *(defined only for MBRW) and SD are community average connectivity, circulation number and standard deviation, respectively. The table compares performance of the memory-biased random walker (MBRW), topologically biased random walk (TBRW) and Girvan–Newman, algorithms*

			Correctly classified nodes in % (SD)		
k_{int}	k_{com}	C_n	MBRW	TBRW	Girvan–Newman
1	15	1	100 (0)	97.97 (0.64)	100 (0)
2	14	1	100 (0)	96.25 (0.51)	100 (0)
3	13	2	99.7 (0.66)	91.72 (0.75)	99.9 (0.003)
4	12	2	98.4 (0.74)	89.53 (11.7)	97.3 (0.079)
5	11	3	96.6 (1.66)	85.39 (6.2)	87.3 (0.16)
6	10	3	93.6 (2.38)	75.24 (9.1)	50.6 (0.28)
7	9	4	72.0 (1.37)	66.17 (7.9)	31.88 (0.1)
8	8	4	61.1 (5.24)	58.28 (6.6)	27.3 (0.04)

3. Results

3.1 *Detection of community structure via analysis of memory-biased movement patterns*

3.1.1 *Comparative performances for the Girvan–Newman networks.* As we have previously shown, MBRW leads to preferred motion in regions of the network that were more intra than inter connected [34]. Here, we show how the tendency of the MBRW to inhabit community-like structures can be used to detect communities in different networks. Searching Girvan–Newman networks for communities we found that MBRW algorithm provides results that are very similar to those of the Girvan–Newman's divisive algorithm. MBRW correctly classifies over 98% of the nodes when $k_{int} \leqslant 4$, where k_{int} is the inter-community link number per node. For larger values of k_{int} ($k_{int} \geqslant 5$), MBRW outperforms the Girvan–Newman method (Table 1). This finding is quite important since communities in real-world networks are often highly ambiguous making the case of high k_{int} most common [35]. When compared with spectral clustering, application of the MBRW method improves community detection, a fraction of correctly classified nodes, for all values of k_{int} (note the standard deviation column in Table 1).

3.1.2 *Performance for the cored networks.* We next tested the relative ability of the MBRW algorithm to detect the cores in the cored networks by comparing its performance to the standard Girvan–Newman and TBRW methods as before. Once again the MBRW method outperformed existing methods by large margin. In fact, arguably due to the cored networks' uneven structure, MBRW outpaces all other methods for $k_{int} > 1$ (Table 2).

3.1.3 *MBRW performance on the realistic benchmark networks (17).* We next compared the two community detection algorithms, MBRW (run for up to $C_n = 4$) and Infomap, on a set of benchmark networks [17] with different mixing parameters μ (Table 3). In all of these networks, the number of communities embedded by design in the 10 instances of the benchmark networks M_{n1} was between with an average of 80.4 ± 13.8 (see the community number band of the designed communities in Fig. 4).

TABLE 2 *Correctly classified nodes for the cored networks with different inter-community link numbers per node (k_{int}). k_{com}, C_n (defined only for MBRW) and SD are community average connectivity, circulation number and standard deviation, respectively*

			Correctly classified nodes in % (SD)		
k_{int}	k_{com}	C_n	MBRW	TBRW	Girvan–Newman
1	9	1	100 (0)	100 (0)	100 (0)
2	8	1	97.71 (0.36)	96.1 (0.85)	91.68 (2.4)
3	7	2	94.72 (0.35)	93.84 (0.32)	66.95 (3.58)
4	6	2	74.04 (2.12)	55.78 (2.9)	38.03 (2.89)
5	5	3	41.52 (1.39)	33.32 (1.67)	21.43 (0.19)

TABLE 3 *Comparison of the mean module densities with their associated community numbers obtained by the MBRW (M_{n1}, M_{n2} and M_{n3}) and Infomap (M_{n2}) algorithms on the benchmark networks [17] of $N = 5000$ with different mixing rates (μ). Mean module densities of the designed communities (the By Design column) are listed in the table for reference. Note that these densities are the mean module densities of the exact communities as placed in the networks by design. SDs for the module densities and community numbers were in the range of 0.004–0.025 and 3.69–13.32, respectively*

	Mean module density				
	MBRW			Infomap	By Design
μ	(M_{n1})	(M_{n2})	(M_{n3})	(M_{n2})	(M_{n1})
0.1	0.416	0.416	0.439	0.416	0.420
0.2	0.355	0.343	0.387	0.338	0.357
0.3	0.330	0.304	0.358	0.297	0.330
0.4	0.267	0.226	0.337	0.212	0.264
0.5	0.233	0.199	0.295	0.180	0.228

Infomap identifies the optimal number of communities, M_{n2} and partitions the network only for this value. Interestingly, M_{n2} wasn't only consistently larger than the corresponding M_{n1}, but it grew larger with the rise in μ. Using MBRW we could easily estimate communities for different community numbers and show the mean module density for each M_n. We identified an optimum MBRW community number M_{n3} by applying MBRW across a wide range of community numbers M_{n1} and M_{n2} for the networks with a range of mixing parameters μ.

The MBRW optimal community number M_{n3} was consistently smaller than the related M_{n1}, growing smaller with the rise in μ (Fig. 4). At all values of the mixing parameter μ, the communities identified by MBRW outperformed Infomap in terms of module density when either M_{n1} or M_{n2} community numbers were chosen (Table 3 and Figs 5–7). Furthermore, the module densities for community numbers M_{n3} were higher still (Table 3 and Figs 8 and 9).

Going beyond the mean module densities, we compared the differences in the module densities of the individual communities at M_{n1} and M_{n2} community numbers, as identified by MBRW to their module densities of communities as designed or obtained by Infomap, respectively.

To compare, we ranked the individual community module densities of both community sets (either MBRW vs. Designed or MBRW vs. Infomap set) by their module density and then subtracted them from

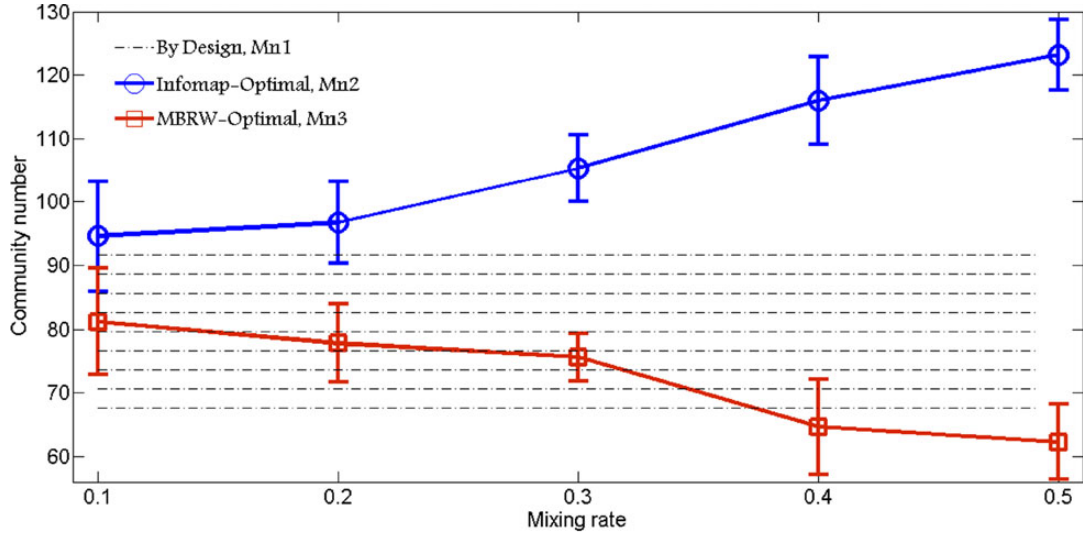

FIG. 4. Comparison of the community numbers obtained by the MBRW and Infomap algorithms on the benchmark networks [17] of $N = 5000$ with different mixing rates (μ). The band of the designed number of communities (dashed area in the figure) is given for reference. Community numbers used for producing this figure are the 10-run average community number for the actual network (M_{n1}), and the optimums obtained by Infomap (M_{n2} circles in blue) and MBRW (M_{n3} squares in red).

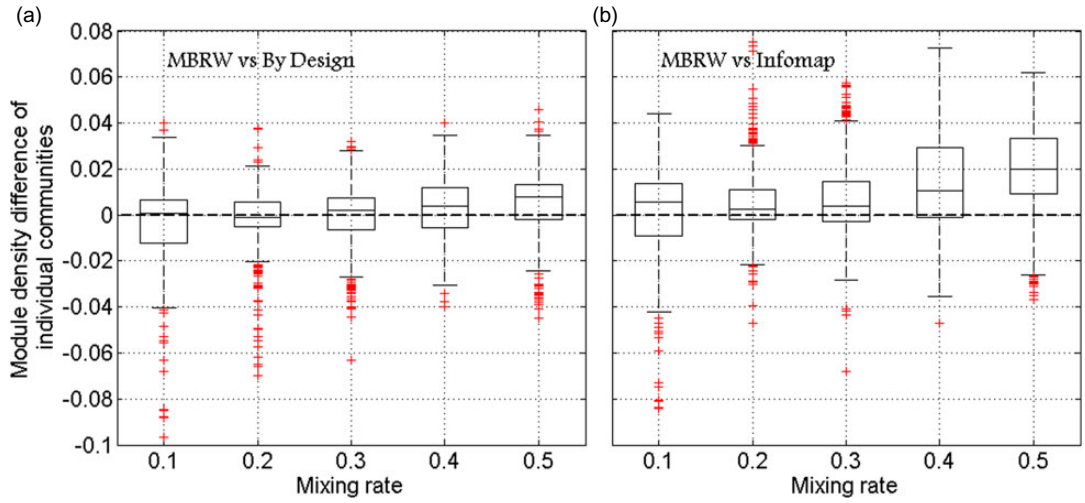

FIG. 5. Distribution of the difference between ranked module densities at different mixing rates for: (a) M_{n1} communities identified by MBRW and by design, and (b) M_{n2} communities identified by MBRW and by Infomap. Lines in boxes show median difference, boxes represent the 25th and 75th percentiles while the region between whiskers represent 99.3% of the data for each box-plot.

each other matching on their rank (i.e. the module densities in the same rank of the two different sets are subtracted). Most of if not all of the communities identified by MBRW have higher module density (the values are positive). This trend becomes even more pronounced as μ goes up (Figs. 5–7).

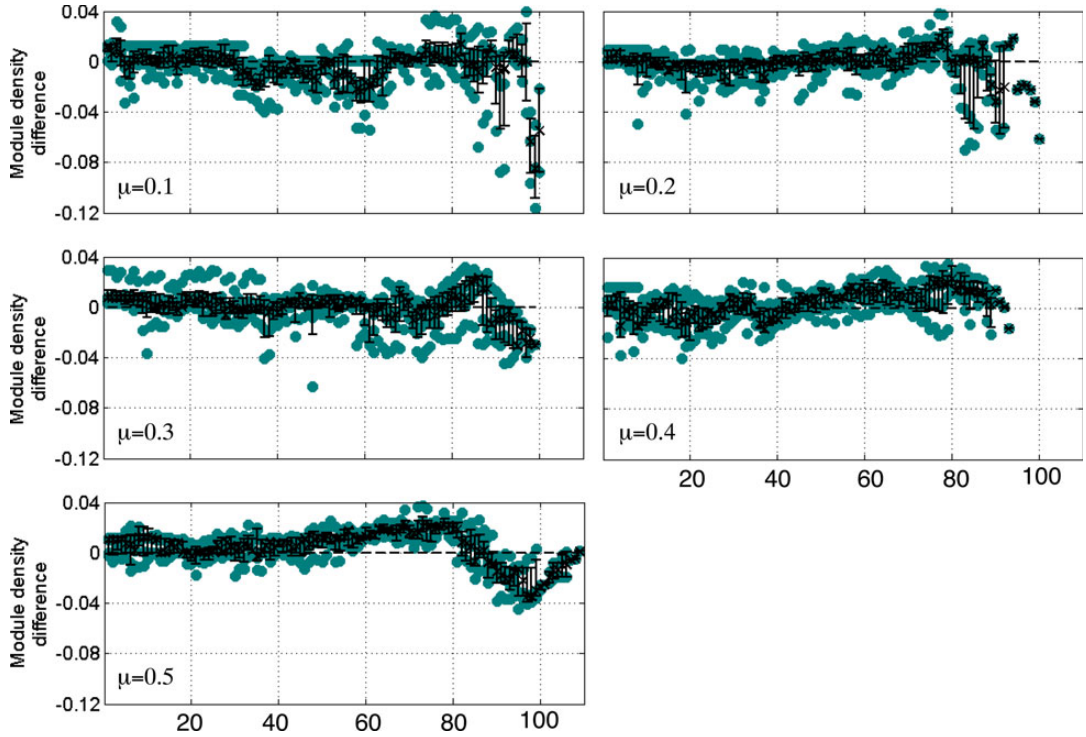

FIG. 6. Differences between M_{n1} ranked module densities of the MBRW and designed communities for the 10 networks simulated (marked by green dots) at few levels of mixing ($\mu = 0.1$ to 0.5). The markers 'x' with whiskers represent the median with first and third quartiles for each module rank across simulations.

We compare the community quality at the individual community level by ranking the M_{n1} and the M_{n2} communities identified by MBRW by their module density and comparing them to the equivalent M_{n1} and M_{n2} communities identified by design [17] or by Infomap, respectively (Figs 6 and 7). The medians indicated by the 'x' marker, suggest that MBRW generates better community structure compared to other algorithms, especially at high mixing levels. For $\mu = 0.1, 0.2, 0.3, 0.4$ and 0.5, the MBRW communities achieved module densities higher than the module densities of the originating M_{n1} designed communities in 44, 52, 58, 61 and 66% of the compared communities, respectively, for the mixing rates (Fig. 6). The improvement is even more pronounced when compared with Infomap: MBRW identified M_{n2} communities in which 61, 64, 72, 76 and 82% of the communities had higher module density compared with Infomap for, respectively, $\mu = 0.1, 0.2, 0.3, 0.4$ and 0.5 (Fig. 7). In general, it appears that MBRW identifies communities with higher module density at the expense of a uniform modular structure. Thus for instance, at the optimal community number of MBRW M_{n3}, the vast majority of the communities have a module density between 5 and 15 times higher than the overall network module density while 2–3 have essentially no community structure and a module density that matches it with a value of ~ 0.004 (Fig. 8). Similarly, when the network is partitioned into M_{n3} communities by MBRW the module densities between the 1st and 3rd quartile follow a much smaller range of module densities which are all quite high but the overall range of module densities remains much the same (Fig. 9).

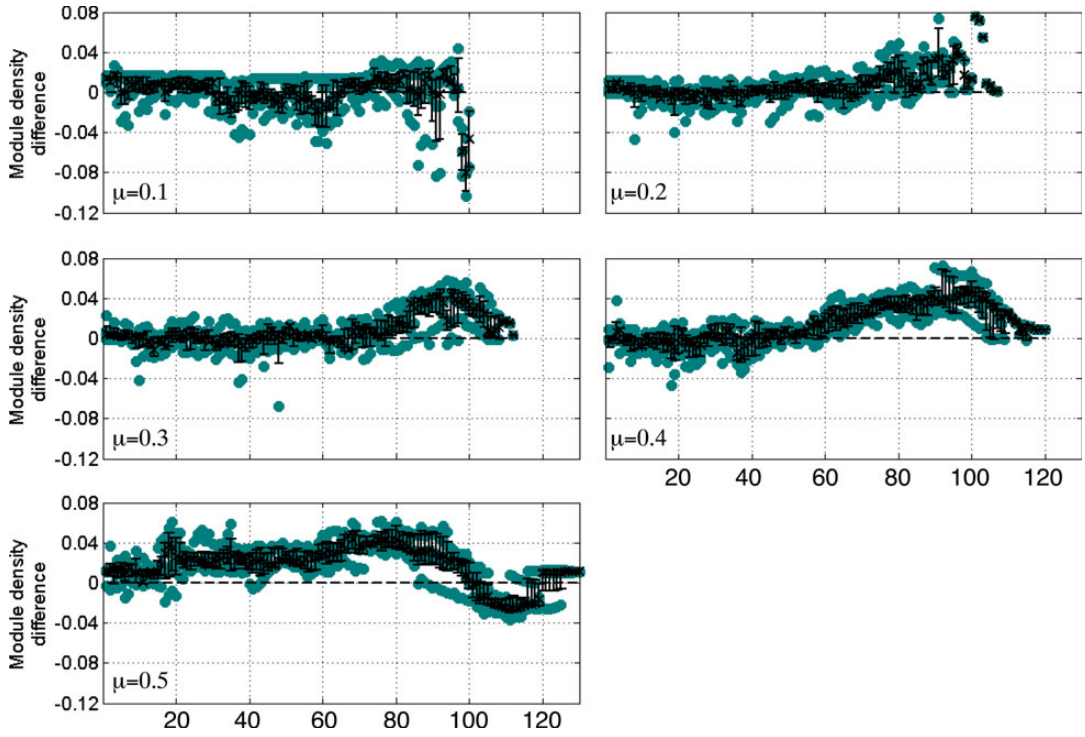

FIG. 7. Differences between M_{n2} ranked module densities of the MBRW and Infomap communities for the 10 networks simulated (marked by green dots) at several levels of mixing ($\mu = 0.1$ to 0.5). The markers 'x' with whiskers represent the median with first and third quartiles for each module rank across simulations.

3.1.4 *Community analysis of the C. elegans protein interaction network.* Finally, we tested the MBRW method on the *C. elegans* protein interaction network. In contrast with the artificial networks where the number of communities is known by design, their number had to be inferred empirically in the real-world networks. As explained in the methods section above, we determined the number of communities in the network by analysing the largest gaps of the consecutive coefficient pairs in the leading eigenvector. The number of candidate communities (M_n) is inferred from the average module density of its corresponding community set and reaches its global maximum around $M_n = 15$ (Fig. 10).

Figure 10 suggests that MBRW network partitioning is optimal for values of around $M_n = 15$, whereas the Infomap optimal community number for the same network is $M_n = 17$. We compared the mean module density of the communities generated by all algorithms—MBRW, Girvan–Newman and Infomap, for $M_n = 10$ to 17 (Table 4).

MBRW again outperforms the traditional Girvan–Newman algorithm for all the tested community numbers from 10 to 17 and Infomap for its optimal $M_n = 17$ (module density = 0.504, Table 4).

3.2 *MBRW runtime performance*

Beyond the ability to infer a realistic community structure, practical use of the network partitioning algorithm is defined by its computational complexity. We test the MBRW efficiency and assess its computational complexity by measuring how the network size effects the mean simulation step numbers

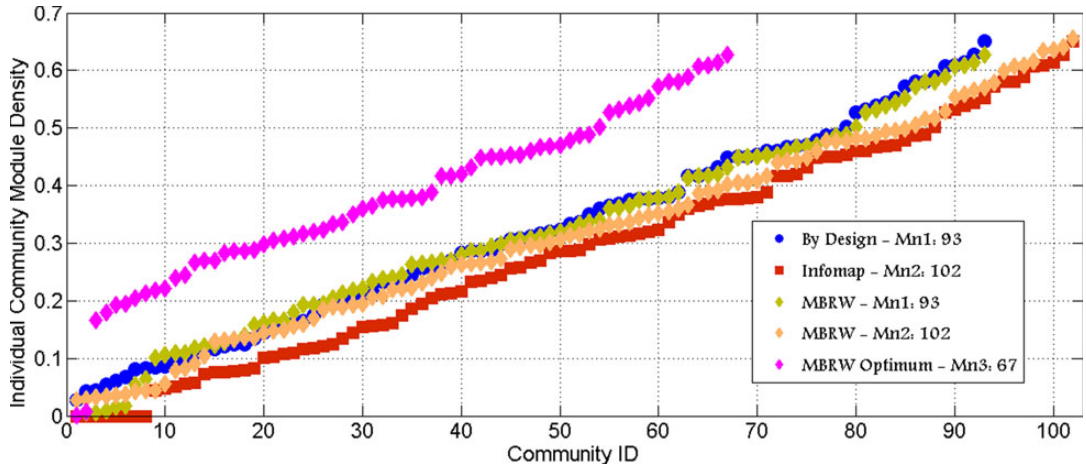

FIG. 8. Ranked module density distributions at $\mu = 0.3$ of the individual communities: by design (blue dots representing $M_{n1} = 93$ communities) by Infomap (red squares representing $M_{n2} = 102$ communities) or by MBRW (green, yellow and pink diamonds representing M_{n1}, M_{n2} and $M_{n3} = 67$ communities respectively).

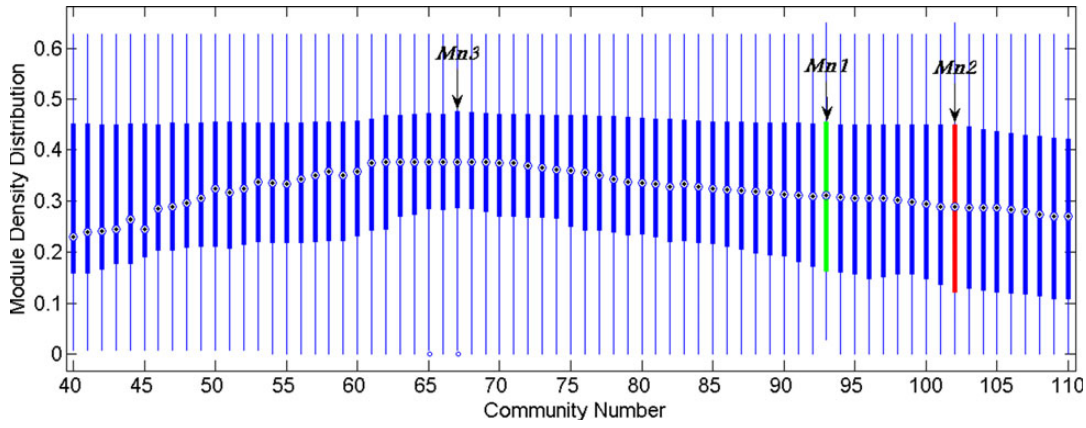

FIG. 9. $N = 5000$, $\mu = 0.3$, 95% confidence interval box-plot of module density distribution identified by MBRW for a range of community numbers. Marked are the optimal results for MBRW ($M_{n3} = 67$) and Infomap ($M_{n2} = 102$ in red) and for the benchmark networks community size by design ($M_{n1} = 93$ in green).

(the sizes of the movement sequences) and the physical times required by the MBRW simulations to converge (Table 5). To be comparable, these measurements are performed for a circulation number of 1 ($C_n = 1$), which we consider to be the basic computational unit. Increase in C_n in the expectation of the MBRW conversion will not alter the functional dependence of the algorithm performance on the network size, making it particularly convenient unit of measurement. Note that MBRW performance scales linearly with parallelization as one can execute numerous instances of the stochastic algorithm independently (e.g. several moving instances traversing the same network, each in separate computation thread or even on a different machine).

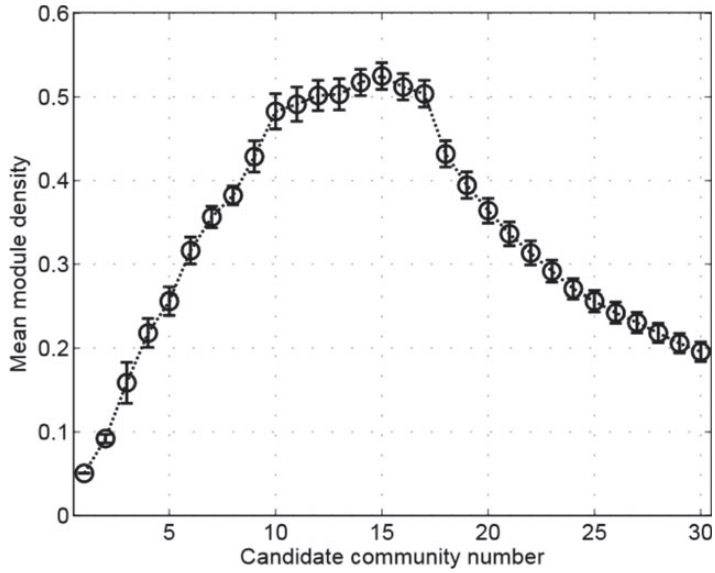

FIG. 10. Average of the mean module density in the *C. elegans* protein interaction network over 25 MBRW simulations with the leading eigenvector coefficient gap analysis for a range of candidate community numbers M_n. Error bars represent SD.

TABLE 4 *Mean module density of the communities identified in the C. elegans protein interaction network with different methods for different community numbers* (M_n). *Note that Infomap generates partition for an Infomap-optimal partition number only, which for this network is* $M_n = 17$

	Mean module density		
M_n	MBRW	Girvan–Newman	Infomap
10	0.483	0.401	–
11	0.492	0.416	–
12	0.502	0.418	–
13	0.503	0.388	–
14	0.517	0.363	–
15	0.525	0.341	–
16	0.512	0.323	–
17	0.504	0.307	0.465

Not surprisingly, the simulation step number per node is independent of the network size (Table 5). On the other hand, the computation time per node grows linearly with the size of the network (Fig. 11) remaining reasonable even for fairly large networks ($\sim$ 7 min for analysis of a 100,000 node network – Table 5).

The identified relationship between physical time per node (computed as computation time/N) and the network size N (Fig. 11) implies that the computational time is parabolic to N, suggesting computation complexity of $O(N^2)$. Although we have obtained a general linear relationship between the network size and physical time per node, the parameters of the line equation to be derived will depend

TABLE 5 *Mean step numbers (per node) and physical times that take a memory-biased random walker to visit all the network nodes at least once as a function of the network size (N). MBRW was executed with a circulation number of $C_n = 1$ on a set of benchmark networks described in Section 2 [17]*

$N(\times 10^4)$	Step number (per node)	Physical time (s)
1	20.96 (3.99)	4.7 (1.06)
2	19.33 (2.18)	15.7 (2.06)
3	20.66 (4.38)	39.1 (6.29)
4	19.14 (2.61)	65.7 (8.96)
5	20.39 (4.12)	106 (21.78)
6	20.79 (3.73)	157.8 (28.35)
7	21.4 (2.52)	222.5 (26.3)
8	20.27 (3.69)	308.8 (48.13)
9	21.2 (4.33)	356.4 (72.94)
10	20.57 (3.27)	439.9 (66.76)

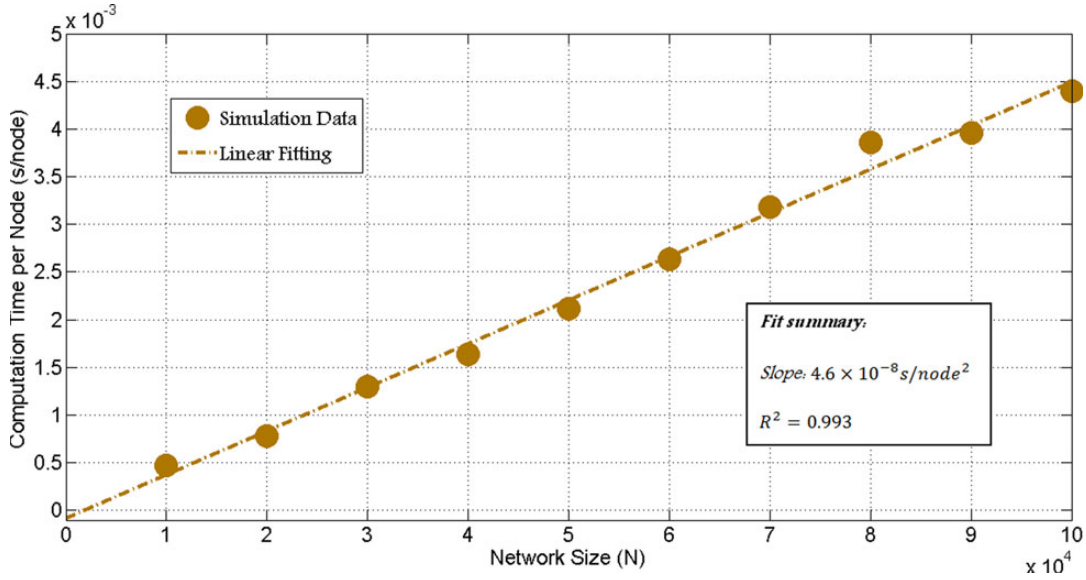

FIG. 11. Physical time per node with respect to the network size (circles, lineage fit represented by dashed line).

on the algorithm implementation details, topological properties of the networks and the workstation configuration.

4. Discussion and conclusion

In the previous work, we showed that introduction of memory bias into otherwise random walk on network causes repetitive movement patterns that reflect structures that are more intra-connected than inter-connected with the rest of the network [34]. In this work we use a memory biased motion algorithm to develop a hybrid bottom-up spectral methodology for community detection. Memory-induced biases

in movement of an entity on network result in construction of an empirically weighted transition matrix which we subject to spectral analysis to detect the network community structure. We test our approach on several artificial (Girvan–Newman (3), cored and benchmark [17]) as well as a real-world (*C. elegans* protein interaction [22]) networks. In the Girvan–Newman network, the MBRW produced similar or better results (especially in poorly defined communities, with $k_{\text{int}} > 5$) than the Girvan–Newman's divisive algorithm or spectral clustering of the network topology (TBRW) [4,6]. We further define cored networks—a new class of random network models exhibiting tuneable community structure, which is less pronounced than in the classical models. Cored networks are convenient for testing performance of community detection algorithms on networks with weakly pronounced, ambiguous community structure that resembles real-world situations. Networks generated by the cored networks model embed densely interconnected structure into a larger network. This is in contrast to Girvan-Newman type networks in which uniform communities are separated from one another with relatively clear boundaries. Using spectral analysis applied to transition matrix, we define module density that is instrumental in assessing the quality of the detected community structure. In terms of module density, MBRW outperformed the Girvan–Newman and spectral clustering methods and achieved efficient partitioning for the cored networks even with highly ambiguous cores.

In addition to Girvan–Newman and spectral analysis methods, we compared performance of MBRW to that of the Infomap algorithm which shares the underlying operation principle with the Louvain's method. Both Infomap and MBRW are flow-based and both rely on the analysis of a random walker's movement in a given network. The main difference between Infomap and related methods and MBRW is that while the former use a theoretical stationary distribution of node visits of the walker, MBRW effectively traces the *actual* movement of a walker and constructs the communities out of this movement. Furthermore, MBRW employs memory to direct the otherwise random network traversal. More specifically, the memory effects movement in two ways: (1) the walker does not retrace its immediate last step and (2) it remembers its last s steps and tends to follow the route decisions as it did within those steps. Infomap employs multiple-order Markov processes as well (in which non-backtracking rule does not apply). But Infomap and MBRW differ in how they define and implement memory. In Infomap, the memory is path-dependent so that the walker's next move depends on where it comes from. In contrast, the MBRW memory is not necessarily path-dependent: the next step depends on the frequency of recent visits rather than the exact sequence of steps and the current location. The MBRW walker is attracted toward the nodes that have already been visited. We found that the MBRW emphasis on the analysis of a transient state is superior to the Infomap emphasis on the analysis of a stationary state in achieving efficient identification of communities in the benchmark networks of [17] (Table 3 and Figs. 5–9).

Furthermore, across the whole range of mixing parameter (μ) when we use MBRW for the different optimal partition numbers as described by the design of the benchmark network, by Infomap or by MBRW (M_{n1}, M_{n2} and M_{n3}, respectively) MBRW performance is either equivalent or superior to other algorithms (Table 3, Figs. 5 and 8). In all cases, Infomap tends to divide networks into higher number of (smaller) communities and suggests more communities than were put in by design, while MBRW find equivalent number of communities at lower mixing ($\mu = 0.1 - 0.3$) and fewer when mixing rises (Fig. 4).

To fully compare MBRW and Infomap, we heuristically derived and analysed the optimum community numbers obtained by MBRW that give the highest mean module densities (Table 3, Figs. 8 and 9). MBRW identifies sets of communities whose mean module densities are $\sim$5–30% higher than the mean module densities of the designed communities for the networks with the clearest ($\mu = 0.1$) and most ambiguous ($\mu = 0.5$) communities, respectively (Table 3). Here, it may seem contradictory

for MBRW uncovering community sets whose mean module density is superior (if not equal) than the sets put in the networks by design especially for the networks with ambiguous community profile (i.e. higher μ). Considering the creation procedure of the benchmark networks from [17] in which the intra/inter-links are added to the networks at random, it is always possible to have a configuration where any two groups of nodes (assumingly candidate communities) will be relatively more inter-connected than intra-connected for higher μ. Such a case would justify the MBRW algorithm's merging of the two groups. In fact, this was exactly how MBRW partitioned the benchmark networks (Table 3). It tended to merge the loose communities with dense inter-connections obtaining fewer, yet better-defined partitioning (Figs. 4, 8 and 9). For example, although there were on average 80.4 communities put by design in the networks with $\mu = 0.5$ and mean module density of 0.228, MBRW identified fewer communities (62.3 on average) but with a higher mean module density (0.295) (Table 3). Possibly, the higher module density for the M_{n3} MBRW optimal communities is achieved by removing some parts of the network from having any community structure at all. For instance, we observe that 61 of the 67 optimal communities identified by MBRW have module density > 0.2 while the two lowest ranking MBRW communities have a module density of ~ 0.004, which is slightly lower than the overall module density of the containing network (0.0043) (Fig. 9).

MBRW also outperforms Infomap in the real-world *C. elegans* protein interaction network, yielding significantly higher nodular density for 15 communities vs. 17 for Infomap (Table 4). Despite the absence of consensus on the definition of a network community, many approaches share the notion that communities are defined as relational concepts [26]. We adopted a similar approach in this work and compared the mean module density of the local structures to the module density for the entire network. This measure defines the community status of the set of local structures. Thus, 15 candidate communities yielding a mean module density of ~ 11 times higher than those of the entire network represent a very reasonable segmentation (Fig. 10 and Table 4).

Contemporary researchers routinely use network abstraction to represent large biological and social systems. These systems' structural modules tend to have some functional autonomy, while interacting with other system's elements as they remain embedded in the network at different levels. Understanding the system organization can therefore be seen as network partitioning problem. In spite of the need, the currently available algorithms are insufficient while developing algorithms that efficiently reveal the community structure of real networks is still highly challenging. We believe that the algorithm presented here can be seen as an early example of a dynamical bottom-up approach to explore the community-like structures utilizing a process that may, in fact, resemble processes occurring on the network. Indeed, beyond being a network analysis tool, the MBRW presented here could owe its performance to the fact that it catches some of the behaviours in real world biological and social information networks. Entities traversing such systems rarely have access to maps of the whole network and substitute such global knowledge by gradual exploration. This idea is further supported by the fact that our method significantly outperforms alternative approaches, particularly in networks with ambiguous communities and in the cases when network communities are not uniform but form 'cores' embedded in the network. We believe that the MBRW algorithm represents a more realistic and applicable approach to partitioning of the real world networks.

Funding

Mesut Yucel was funded by the Scientific and Technological Research Council of Turkey (TUBITAK) fellowship. Uri Hershberg is a Stein fellow and the collaboration in this work was supported in part by the the Louis and Bessie Stein Family Fellowship for Exchanges with Israeli Universities.

References

1. Barabasi, A. L. & Oltvai, Z. N. (2004) Network biology: understanding the cell's functional organization. *Nat. Rev. Genetics*, **5**, 101.
2. Alm, E. & Arkin, A. P. (2003) Biological networks. *Curr. Opin. Struct. Biol.*, **13**, 193–202.
3. Girvan, M. & Newman, M. E. J. (2002) Community structure in social and biological networks. *PNAS*, **99**, 7821–7826.
4. Zlatić, V., Gabrielli, A. & Caldarelli, G. (2010) Topologically biased random walk with application for community finding in networks. *Phys. Rev. E*, **82**, 066109.
5. Newman, M. E. J. (2006) Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, **74**, 036104.
6. Zhou, H. & Lipowsky, R. (2004) Network Brownian motion: a new method measure vertex–vertex proximity and to identify communities and subcommunities. ICCS 2004 (M. Bubak *et al.* eds), LNCS 3038, pp. 1062–1069.
7. Lancichinetti, A. & Fortunato, S. (2009) Community detection algorithms: a comparative analysis. *Phys. Rev. E*, **80**, 056117.
8. Clauset, A., Newman, M. E. J. & Moore, C. (2004) Finding community structure in very large networks. *Phys. Rev. E*, **70**, 066111.
9. Danon, L., D'iaz-Guilera, A., Duch, J. & Arenas, A. (2005) Comparing community structure identification. *J. Stat. Mech.: Theory Exp.*, **09**, P09008.
10. Shen, H.-W. & Cheng, X.-Q. (2010) Spectral methods for the detection of network community structure: a comparative analysis. *J. Stat. Mech.: Theory Exp.*, **10**, P10020.
11. Chauhan, S., Girvan, M. & Ott, E. (2009) Spectral properties of networks with community structure. *Phys. Rev. E*, **80**, 056114–00.
12. Rosvall, M., Esquivel, A. V., Lancichinetti, A., West, J. D. & Lambiotte, R. (2014) Memory in network flows and its effects on community detection, ranking, and spreading. *Nat. Commun.* doi:10.1038/ncomms5630
13. Rosvall, M. & Bergstrom, C. (2008) Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. U.S.A.*, **105**, 1118.
14. De Domenico, M., Lancichinetti, A., Arenas, A. & Rosvall, M. (2015) Identifying modular flows on multilayer networks reveals highly overlapping organization in interconnected systems. *Phys. Rev.*, **5**, 011027.
15. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. (2008) Fast unfolding of communities in large networks. *J. Stat. Mech.: Theory Exp.*, **2008**, P10008.
16. Ikeda, S., Kubo, I. & Yamashita, M. (2009) The hitting and cover times of random walks on finite graphs using local degree information. *Theor. Comput. Sci.*, **410**, 94–100.
17. Lancichinetti, A., Fortunato, S. & Radicchi, F. (2008) Benchmark graphs for testing community detection algorithms. arXiv:0805.4770.
18. Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N. & Barabasi, A.-L. (2000) The large-scale organization of metabolic networks. *Nature*, **407**, 651.
19. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D. & Alon, U. (2002) Network motifs: simple building blocks of complex networks. *Science*, **298**, 824–827.
20. Flake, G. W., Lawrence, S., Giles, C. L. & Coetzee, F. M. (2002) Self-organization and identification of web communities. IEEE. 0018-9162/02
21. Guimera, R., Mossa, S., Turtschi, A. & Amaral, L. A. N. (2005) The worldwide air transportation network: anomalous centrality, community structure, and cities' global roles. *Proc. Natl. Acad. Sci. U.S.A.*, **102**, 7794–7799.
22. Gunsalus, K. C., Ge, H., Schetter, A. J., Goldberg, D. S., Han, J.-D. J., Hao, T., Berriz, G. F., Bertin, N., Huang, J., Chuang, L.-S., Li, N., Mani, R., Hyman, A. A., Sönnichsen, B., Echeverri, C. J., Roth, F. P., Vidal, M. & Piano, F. (2005) Predictive models of molecular machines involved in *Caenorhabditis elegans* early embryogenesis. *Nature*, **436**, 861–865.

23. GILLESPIE, D. T. (2001) Approximate accelerated stochastic simulation of chemically reacting systems. *J. Chem. Phys.*, **115**, 1716.

24. GILLESPIE, D. T. (1977) Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, **81**, 2340–2361.

25. LIPOWSKI, A. & LIPOWSKA, D. (2012) Roulette-wheel selection via stochastic acceptance. *Phys. A: Stat. Mech. Appl.*, **391**, 2193–2196.

26. LESKOVEC, J., LANG, K. J. & MAHONEY, M. W. (2010) Empirical comparison of algorithms for network community detection. *Proceedings of WWW'10*, pp. 631–640.

27. NEWMAN, M. E. J. (2006) Modularity and community structure in networks. *PNAS*, **103**, 8577–8582.

28. BENDER, E. A. & CANFIELD, E. R. (1978) The asymptotic number of labelled graphs with given degree sequences. *J. Comb. Theory Ser. A*, **24**, 296–307.

29. FORTUNATO, S. & BARTHÉLEMY, M. (2007) Resolution limit in community detection. *Proc. Natl Acad. Sci.*, **104**, 36–41.

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

31. KANUNGO, T., MOUNT, D. M., NETANYAHU, N. S., PIATKO, C. D., SILVERMAN, R. & WU, A. Y. (2002) An efficient k-means clustering algorithm: analysis and implementation. *IEEE Trans.*, **24**, 881–892.

32. ROSVALL, M. & BERGSTROM, C. T. (2010) Mapping change in large networks. *PLoS One*, **5**, e8694.

33. BOLLOBAS, B. (1984) *Graph Theory and Combinatorics: Proceedings of the Cambridge Combinatorial Conference in Honour of Paul Erdös (21–25 March 1983)*. Cambridge: Academic Press.

34. YUCEL, M. & HERSHBERG, U. (2014) Memory as an organizer of dynamic modules in a network of potential interactions, SCW at AAMAS.

35. LESKOVEC, J., LANG, K. J., DASGUPTA, A. & MAHONEY, M. W. (2008) Statistical properties of community structure in large social and information networks. *Proceedings of WWW'08*, pp. 695–704.