Community structure identification, A modern review

Leon Danon and Albert Díaz-Guilera

Departament de Fisica Fonamental, Universitat de Barcelona,

Marti i Franques 1 08086 Barcelona, Spain

Jordi Duch and Alex Arenas

Departament d'Enginyeria Informàtica i Matemàtiques,

Universitat Rovira i Virgili, 43007 Tarragona, Spain

PACS numbers:

I. INTRODUCTION

The study of complex networks has received an enormous amount of attention from the scientific community in recent years[1–5]. Physicists in particular have become interested in the study of networks describing the topologies of wide variety of systems, such as the world wide web, social and communication networks, biochemical networks and many more. Although several questions have been addressed, many important ones still resist complete resolution. One such problem is the analysis of modular structure found in many networks [6]. Distinct modules or communities within networks can loosely be defined as subsets of nodes which are more densely linked, when compared to the rest of the network. Such communities have been observed, using some of the methods we shall go on to describe, in many different contexts, including metabolic networks [7, 8], banking networks [9] and most notably social networks [10]. As a result, the problem of identification of communities has been the focus of many recent efforts.

Community detection in large networks is potentially very useful. Nodes belonging to a tight-knit community are more than likely to have other properties in common. In the world wide web, community analysis has uncovered thematic clusters [11–13]. In biochemical or neural networks, communities may be functional groups[14], and separating the network into such groups could simplify the functional analysis considerably.

The problem of community detection is particularly tricky and has been the subject of discussion in various disciplines. A simpler version of this problem, the graph bipartitioning problem (GBP) has been the topic of study in the realm of computer science for decades. Here one looks to separate the graph into two densely connected communities of equal size, which are connected with the minimum number of links. This is an NP complete problem [15], however several methods have been proposed to reduce the complexity of the task [16–18]. In real complex networks we often have no idea how many communities we wish to discover, but in general it is more than two. This makes the process all the more costly. What is more, communities may also be hierarchical, that is communities may be further divided into sub-communities and so on [19–21].

In this review we would like to present the recent advances made in the field of community identification in networks in a clear and simple fashion. To this end, the sections are organised as follows. In the next section we describe some ways to define communities

in a network context. Following this, we present a method to evaluate the a particular partition of a network. Then, we go on to descibe the various recent methods starting with link removal methods, going on to agglomerative methods, followed by methods optimising modularity and finally "other" methods. Some of the methods presented do not necessarily fit into just one of these classification, and there may be some overlap.

II. DEFINITIONS OF COMMUNITIES

Despite the large amount of study in this area, a consensus on what is the definition of community has not been reached. With a few exceptions, we will mostly be dealing with networks in which the links have no direction and are unweighted. In this case the definition of community must be purely topological.

Social scientists have been treating the subject of social networks for some time. For a standard text on the social science approach to networks analysis see [22]. Their approach is largely (though by no means exclusively) concerned with the effect an individual player has on the network and vice versa. As a result, the local properties of networks take a more prominent role in the social science research. However much of this knowledge is extremely useful. Some definitions taken from [22] have been used and developed by methods we shall describe later. Here we present some of these.

Conceptually, the definitions can be separated into two main categories, self-referring and comparative definitions. Central to all such definitions is the concept of subgraph.

1. **Self referring definitions**

   The basic community definition is *a clique*, defined as a subgroup of a graph containing more than two nodes where all the nodes are connected to each other by means of links in both directions. In other words, this is a fully connected subgraph. This is a particularly strong definition and rarely fulfilled in real sparse networks for larger groups. *n-cliques, n-clans* and *n-clubs* are similar definitions designed to relax the above constraint, while retaining its basic premise. The shortest path between all the nodes in a clique is unity. Allowing this distance to take higher values, one arrives at the definition of *n-cliques*, which are defined as a subgroups of the graph containing more than two nodes where the largest shortest path distance between any two nodes in the group is n. *n-clans* and *n-clubs* are subtle variations of *n-cliques*.

2. **Comparative definitions**

Above, a community has been defined only in reference to itself. A somewhat different approach to this is to compare the number of internal links to the number of external links, coming from the intuitive notion that a community will be denser in terms of links than its surroundings. One such definition, an *LS set* is defined as a set of nodes in which each of its components has more links to other components within the same community. This is the same definition as the *strong definition of community* in [23].

Again the above definition is quite restrictive, and in order to relax the constraints even further, Raddichi *et al.* propose to use the *sum* of links. So a community in the *weak* sense is defined as a set of nodes whose total number of internal links is greater than the total number of links to the outside. This is the most intuitive of all definitions and is the one that is used most, although implicitly.

Self-referring definitions, while useful in characterising communities which are already known, are not the best choice while trying to find them. The Bron-Kerbosch algorithm [24] for finding cliques in a network is very costly, running in worst case time that scales exponentially with network size. Comparative definitions, on the other hand, lend themselves much more easily to the search for communities in large complex networks. In a way, comparing the internal structure of a community to the external structure gives rise to a measure of how good a particular partition is, as described in the next section.

III. EVALUATING COMMUNITY IDENTIFICATION

A question that has been raised in recent years is how to evaluate a given partition of a network into communities. A simple approach that has become widely accepted was proposed in [25]. It is based on the intuitive idea that random networks do not exhibit community structure. Let us imagine that we have an arbitrary network, and an arbitrary partition of that network into N_c communities. It is then possible to define a $N_c \times N_c$ size matrix $\mathbf{e}$ where the elements e_{ij} represent the fraction of total links starting at a node in partition i and ending at a node in partition j. Then, the sum of the any row (or column) of $\mathbf{e}$, $a_i = \sum_j e_{ij}$ corresponds to the fraction of links connected to i.

If the network does not exhibit community structure, or if the partitions are allocated

without any regard to the underlying structure, the expected value of the fraction of links within partitions can be estimated. It is simply the probability that a link begins at a node in i, a_i, multiplied by the fraction of links that end at a node in i, a_i. So the expected number of intra-community links is just $a_i a_i$. On the other hand we know that the *real* fraction of links exclusively within a partition is e_{ii}. So, we can compare the two directly and sum over all the partitions in the graph.

$$Q = \sum_i (e_{ii} - a_i^2) \tag{1}$$

This is a measure known as *modularity*. For a perfect partition, for example two communities corresponding exactly to two disconnected parts of the network, with no links between them, the value of modularity is 1. For particularly bad partitions, for example, when all the nodes are in a community of their own, the value of modularity can take negative values.

One might be tempted to think that random networks will exhibit very small values of modularity. As Guimerà *et al.* show, this in general is not the case [26]. It is possible to find a partition which not only has a nonzero value of modularity for random networks of finite size, but that this value is quite high, for example a network of 128 nodes and 1024 links has a maximum modularity of 0.208. This suggests that these networks that seem to have no structure actually exhibit community structure due to fluctuations.

IV. LINK REMOVAL METHODS

Intuitively, the simplest way to partition a network is to cut some links until the network is no longer connected. Divisive methods do just that. However, cutting links haphazardly is unlikely to give useful results. So, several methods have been proposed to find the most appropriate links to remove, so that the disconnected components correspond to meaningful communities.

A. Shortest path centrality

This is one of the first methods presented, and remains one of the more elegant [10]. It is based on the idea of centrality, a measure of how central the node or link is in the network. Shortest path centrality is employed in this case, and is measured as the number

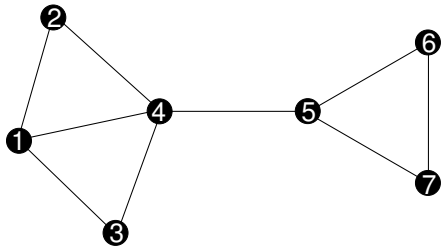


FIG. 1: Shortest path centrality (betweenness) is the number of shortest paths that go through a link or node. In this simple case, the link with the largest link centrality is that joining nodes 4 and 5.

of shortest paths between pairs of nodes that pass through a certain node or link. Intuitively, links which are most central are also the most "between", and as such, will act as bridges joining communities together in a connected whole. Removing these bridges should split the network into more densely connected communities, see Fig. 1.

The algorithm proceeds as follows:

1. Calculate shortest path centralities for all links.
2. Remove link with the highest centrality.
3. Recalculate all link centralities.
4. Repeat from step 2 until the network is split into two parts.
5. Proceed iteratively within each of the partitions until no links remain and the network is reduced to individual nodes, each in its own partition.

Should a particular link removal split the network, it necessarily does so into two components, since a link by definition connects two nodes. As the algorithm proceeds separating the network into ever smaller pieces, it is possible to construct a a dendrogram or *binary tree* to store the information of the entire process for later retrieval and analysis.

Calculation of link betweenness is the most computer intensive part of the algorithm. Using the fastest methods developed independently by Newman [27] and Brandes [28] and for a network of size n with m links the speed of calculating all link betweenness-es still remains of $O(mn)$ for unweighted networks. Unfortunately, the calculation needs to be repeated each time since once any link is removed, the betweenness of all the other links

is affected. In fact Girvan and Newman report that omitting step 3 leads to "wrong" community detection [10].

This algorithm is quite sensitive and is one of the few able to detect community structure at all levels. Its major drawback is the computational cost. It scales with the number of nodes n and number of links m as $O(m^2 n)$, which limits the size of the graph one can treat with this method to around 10000 nodes (with current desktop computer technology and some patience).

B. Current-flow and random walk centrality

In an extension of the method described above, in [25] the same authors present two other means to detect community structure. The basic method remains the same as above, with the difference being the way in which the link centrality is calculated.

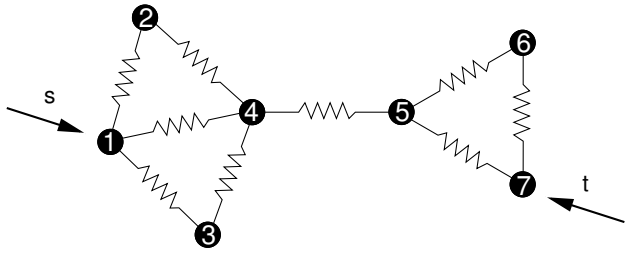


FIG. 2: Resistor networks and current flow centrality. The links in the network are considered as unit resistances. By choosing a pair of nodes to be a source of unit voltage s and sink t, one can can calculate the current flow through any link using Kirchoff's laws. Summing this value for every pair of nodes gives the total current flow betweenness of a link. In this case the biggest current flow is through link joining nodes 4 and 5.

1. *Resistor networks.* In this approach the network to be studied is considered to be a circuit, where links are assigned a unit resistance and a particular pair of nodes act as unit voltage source and sink. The current flows from source to sink along a number of paths, those with the lowest resistance (shortest path) carry the most current. So the *current-flow* betweenness of an link can be calculated using Kirchoff's laws by summing the value of the current flowing through that link over all pairs of nodes. This can be done in only in $O(n^4)$ time for sparse networks since the method is dependent on the inversion of an $N \times N$ matrix which takes $O(n^3)$ time in the worst case.

2. *Random walks.* Here, the network is thought of as a substrate for signals that perform a random walk from a source vertex to a sink vertex. The link betweenness in this case is simply the rate of flow of random walkers through a particular link summed over all pairs of vertices. The authors of [25] show that this measure of betweenness is numerically identical to current flow betweenness, although the derivation is different.

Although conceptually interesting, these approaches are computationally costly. As the authors themselves note, and we can see in Sec. IX, the shortest path betweenness outperforms these approaches in both speed and accuracy. Both the resistor network approach and the random walk approach ideas have been developed further by other authors (see Sec. VIII B and Sec. VII D).

C. Information centrality

Another divisive algorithm was presented by Fortunato *et al.* [29]. In this paper they employ the *network efficiency* measure, previously proposed by Latora and Marchiori [30] to quantify how efficient a particular network $\mathbf{G}$ is in the context of information exchange. Once a particular link is removed from $\mathbf{G}$, its efficiency is reduced by a measurable amount C^I, or *information centrality*. The idea behind the algorithm is that the linkss responsible for the largest drop in network efficiency are those that act as bridges between communities. The algorithm proceeds in a similar way as the GN algorithm, that is, recursively calculating the links with highest C^I and removing them, until the entire hierarchy is unravelled. The algorithm is somewhat slower than other divisive algorithms running at $(O(n^4))$, but what it loses in speed it gains in accuracy. In comparison with the GN algorithm it performs better when the communities to be found are more diffuse, see Sec. IX.

D. Link clustering

This algorithm, proposed in [23] is based on the idea that linked nodes belonging to the same community should have a larger number of 'common friends'. In other words links inside communities should be part of a large proportion of possible loops, and links pointing to outside of the community should be included in few or no loops. The algorithm proceeds as in IV A, but instead of using the link centrality value, it works with the 'link-clustering

coefficient' $C^{(g)}$, which represents the fraction of possible loops of order g that pass through a certain link. The algorithm is implemented for triangles ($g = 3$) or squares ($g = 4$). The system computes the $C^{(g)}$ values for all links, and cuts the one with the *minimum* value. These two steps are repeated recursively as long as all the partitions fulfill one of the community definitions (see Sec. II).

The algorithm is very fast, since calculating the clustering coefficient can be done with local information only. It is also interesting because it was the first algorithm which contained a definition of community to stop the analysis when a certain condition is fulfilled. This method is not appropriate for trees, sparse graphs and disassortative networks due to the small number of triangles and squares.

V. AGGLOMERATIVE METHODS

Instead of starting with the network as a whole and looking for a way to split it into meaningful communities, one can look at the problem from a different perspective. One can start with all the nodes in the network being separate, and use some method to join up, or agglomerate, nodes which are likely to be in the same community.

A. Hierarchical clustering

Traditional methods for detecting communities in social networks have been based on "hierarchical clustering" (see for example [31] and [32]). In general they proceed by calculating a similarity metric for each pair of vertices, representing how close the vertices are according to some property of the network. In the beginning, only vertices in the network are considered, with no links between them, and links are added one by one in order of their weights. Such methods have previously been very successful in small scale case studies, particularly when the complexity of the network under study is not great. Recently however, since this method is very fast and scales well with system size, it has been employed to study the temporal evolution of communities in large networks [33]. Hopcroft *et. al.* study the CiteSeer citation network, where papers in the CiteSeer database are considered as nodes, and citations are considered as links. As a weight metric they use a measure based on the number of citations which any two papers share. The sheer size of this network (around

250,000 papers) makes it intractable with most other methods, and demonstrates the ability of hierarchical clustering methods to deal with large data sets.

B. L-shell method

This method proposes a different take on agglomerative methods. The algorithm proposed in [34] consists of a shell of size l, starting at a node i is a subset of nodes, all within a shortest path distance of $d \leq l$ (L-shell) spreading outward from a starting node i . As the shell expands the *total emerging degree*, K_i^l, is measured which is simply the number of links pointing to vertices outside the expanding shell. When the ratio of the emerging degree at step l to that at step $l-i$, $\frac{K_i^l}{K_i^{l-1}}$, is lower than a cut-off value, the algorithm is stopped. Those nodes within a distance l of the starting vertex are grouped within one community, and all other nodes are said to be outside. This part of the algorithm may be applied when one is concerned with a single community and not the entire community structure, and for this purpose the algorithm is computationally inexpensive scaling linearly with the size of the community under scrutiny. To make the method more general the authors also propose one possible method to apply the algorithm globally. The above process for each node in the network is repeated and a *membership matrix* M is built with the information extracted in the following way: if the process starting at node i classes node j as being in the same community, the element m_{ij} is given the value of 1. Otherwise it is set to 0[55].

VI. METHODS BASED ON MAXIMISING MODULARITY

As described in Sec. III, the modularity measure is one way to evaluate quantitatively a network partition. So, as many authors have asked themselves, why not optimise this value directly? The main problem is that the partition space of any graph (even relatively small ones) is huge, and one needs a guide to navigate this space and find maximum values. Here we outline the approaches that have tackled this problem.

A. Greedy algorithm

In the first attempt at optimising Q directly Newman takes a greedy optimisation (hill climbing) approach [35]. At the start of the algorithm, each node is placed into its own partition. One can then calculate the change in Q should any two partitions be joined. The algorithm proceeds by choosing the pair of partitions producing the largest change, and joining them. This process is repeated until a maximum value of Q is obtained.

The algorithm is one of the fastest available, especially when applied using the data structure for sparse networks described in [36]. However, while also pretty good at identifying community structure, more recent approaches have achieved even more accuracy (see Sec. IX).

This method has been used to study the size distributions of communities. Due to the speed of the approach, large networks can be decomposed into meaningful communities and the distribution of the sizes can be plotted with enough statistics to be able to make conclusions. It has previously been noted that the distribution of community sizes seems to follow a power law [19–21]. However there seems to be some discussion about the exponent of these power laws. The author of [6] shows that for a part of the collaboration network presented in [27], the size distribution follows a power law with exponent -2. However, repeating this experiment we recover an exponent of around -1.5.

B. Simulated annealing methods

Another approach to optimise the modularity measure Q is to employ simulated annealing methods. In [37] the authors present two modifications of the Monte Carlo sampling method with simulated annealing. The process begins with any initial partition of the nodes into communities. At each step, a node is chosen at random and moved to a different community, also chosen at random. If the change improves the modularity it is always accepted, otherwise it is accepted with a probability $\exp(\beta\Delta Q)$[56]. The authors try to improve the success of the method can be improved in two ways. Firstly, the algorithm is stopped periodically, or quenched, and ΔQ is calculated for moving each node to every community that is not its own. Finally, the move corresponding to the largest value of ΔQ is accepted. The second way to improve the efficiency is using a Basin-Hopping approach, where in each step a series

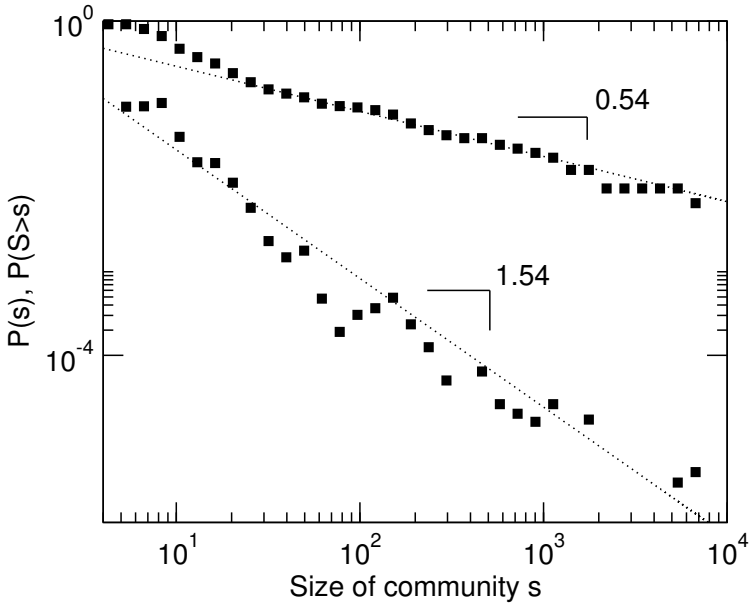


FIG. 3: The size distribution of community sizes in the ArXiv collaboration network. The cumulative distribution follows a power law over several decades with exponent -0.5. The non-cumulative distribution, although noisier seems to follow a power law with exponent with around -1.5 as expected.

of nodes is moved from one community to another, not just one. In this case, the acceptance criterion is calculated directly from the partition that results at the end of the move. The authors report that the second method is slower to run, but able to find high values of Q quickly. In case of large networks it requires less computer memory than the other presented, since it doesn't need extra data structures.

C. Extremal optimisation

In this approach [38], an heuristic search procedure based on extremal optimisation [39] is used to find the network community configuration that has the best modularity value. The algorithm optimises the *local* modularity, a measure which represents the contribution of each node to the *global* modularity. To begin with, nodes are assigned one of two partitions at random. As it evolves, the algorithm improves the contribution of nodes with the worst local modularity, by moving them to the other partition, and therefore improving the global modularity until a maximum possible value is found. Then the links between the two

partitions are removed, and the process is repeated recursively while the modularity keeps increasing.

Extra information is not needed to detect the optimal number of communities and the process stops when the partition modularity cannot be improved further. While not the fastest algorithm, scaling as $O(n^2 log(n))$ (a good application of the greedy algorithm scaling with $O(nlog^2(n))$, it achieves the highest known modularity values for all networks studied, see Sec. IX.

VII. SPECTRAL BASED METHODS

The adjacency matrix of a network contains all the information about the networks topology. A graph of size N can be represented by a matrix A of size $N \times N$ whose element $A_{i,j}$ is zero if no link exists between nodes i and j, and greater than zero if a link exists, where the value of the non-zero element represents the weight of the link.

An alternative representation of an unweighted graph in matrix form is the Laplacian matrix[57]. If a link exists between nodes i and j, the element $L_{i,j} = -1$. The diagonal of the matrix $L_{i,i}$ contains the degree of node i, so that the sum of each row and column is equal to zero. Methods which take advantage of algebraic properties of these matrices have been proposed over several decades.

A. Spectral bisection

Since the sum of elements over each row or column of the Laplacian matrix is equal to zero, there necessarily exists an eigenvector with eigenvalue 0. If the network to be analysed is connected there is only one zero eigenvalue. However, for disconnected graphs with m separate components, the Laplacian matrix is block diagonal, and has m degenerate eigenvectors all corresponding to eigenvalue 0. If the division is not so clear, that is, there exist some links between the m components, the degeneration is no longer present, leaving one eigenvector with eigenvalue zero and $m-1$ eigenvector with eigenvalue slightly greater from zero [40]. So it should be possible to find the blocks, at least approximately by considering the eigenvalues slightly greater than zero and looking at the components of their eigenvectors. As the Laplacian matrix is symmetric, with orthogonal eigenvectors, the sums of the

components of each eigenvector must vanish (apart from the first, trivial eigenvector, which has all equal components). The problem studied in classic papers [17, 41] is a special case, where $m = 2$, the graph bisection problem. Here, the second eigenvector can provide a simple way to cut the graph in two. The components of the *second* eigenvector corresponding to nodes in one subgraph will be positive and so have to be negative for those components corresponding to the other.

Many improvements in both the time it takes for the algorithm to run as well as its precision have been described since. For a review of such methods, specifically for scientific computing see [42].

B. Multi dimensional spectral analysis

Taking further advantage of the properties of the Laplacian matrix, Donetti and Muñoz present a very nice approach in [43]. The first few non-trivial eigenvectors can be extracted sequentially at minimum cost, using the Lanczos method, which can be applied to sparse matrices at minimum computational cost [44]. The individual eigenvector components, which represent nodes in the graph, can be thought of as coordinates in M-dimensional space, where M is the number of non-trivial eigenvectors considered. The idea is that if two nodes belong to the same community, they are close in this M-space. As the authors point out, there is more than one way to measure these distances.

Once separated in this space, the nodes can be clustered using hierarchical agglomerative methods, using both simple Euclidean distance and angular distance. The authors go on to show that the angular version in general performs much better. Once again, the authors employ different methods, both "single linkage" or "multiple linkage" clustering. They show that while faster, single linkage clustering performs worse than the multiple linkage version. The clustering is stopped at the highest value of modularity obtained (see Sec. III), thus detecting the optimal configuration.

This algorithm is reasonably fast ($O(N^3)$according to the authors), but needs *a priori* information on how many vectors need to be extracted to separate the communities properly. In terms of sensitivity, the algorithm performs well (see Sec. IX).

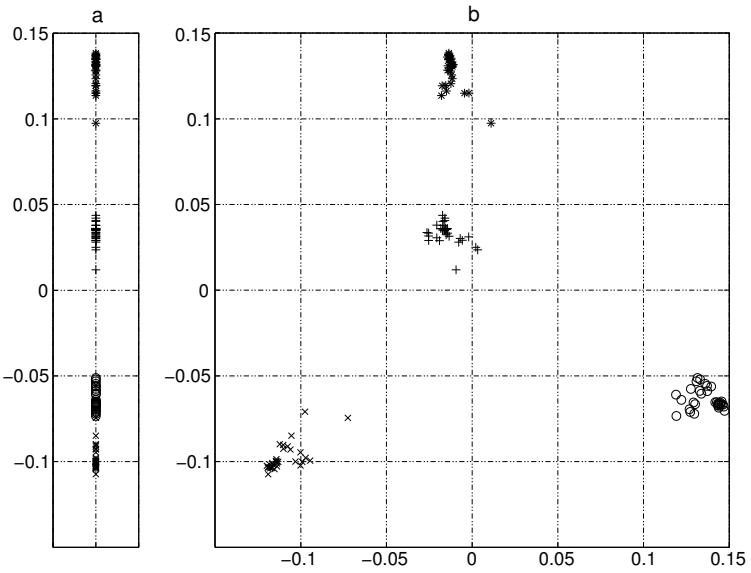


FIG. 4: (a) Components of the first non-trivial eigenvector for a ad. hoc. network with 4 communities (see Sec. IX). (b) All communities can be clearly identified when the components of more than one eigenvector are used as coordinates in M-dimensional space where M is the number of eigenvectors used. Here $M = 2$.

C. Constrained optimisation

This method, described in [12] is based on the spectral properties of the simple adjacency matrix as opposed to the Laplacian. The authors recast the costly problem of extracting eigenvectors of an $N \times N$ matrix into a constrained optimisation problem. In this way they are able to extract the eigenvectors much faster. As in the method of [43] this gives information about the location of the different nodes ordered in different groups in an M-space (where M is once again, the number of eigenvectors extracted). To detect the groups that appear, they use a correlation of the average values of the eigenvectors to measure how close two nodes are in this space. Instead of providing a clear cut community structure, this method gives us an idea of how close any pair of nodes is in the context of communities.

To test the method they study both undirected and directed networks, using the appropriate optimisation function for each case, and test the algorithm on the word association network reported in [45]. The network has over 10000 nodes and the method is able to give qualitatively good results.

D. Approximate resistance networks

In a development of the resistor network approach in [25] Wu *et al.* present an approximate method, in order to reduce the computational time needed [46]. In this method, a pair of nodes is chosen at random to be a voltage source, $V_1 = 1$ and a sinks $V_2 = 0$. The authors then approximate the voltage of all other nodes in the network iteratively, avoiding the costly matrix inversion used in [25]. The accuracy of this approximation is dependent on how many times the iterative step is repeated. After obtaining the node voltages in this way, the values are ordered and large gaps in voltage values are identified. The graph is then split at a particular voltage gap, separating a number of nodes (within a tolerance limit), which must be previously known, from the rest of the network. This process is repeated, randomly choosing pairs of nodes to be voltage sources and sinks. Nodes are then bundled together into a community of the expected size using a simple majority rule over the realisations of the algorithm. Once one community is identified, the process can be restricted to choosing nodes from that community as voltage sources, and sinks from the rest of the network, improving the accuracy.

This method when employed to identify all communities in a graph is dependent on having a good idea of the sizes of communities one is looking for. In networks of larger size and complexity, this is generally not known, and the algorithm becomes more difficult to apply. However, the method can be employed to identify the community that any one nodes belongs to in linear time, similar to the approach of Capocci *et al.* [12], see above.

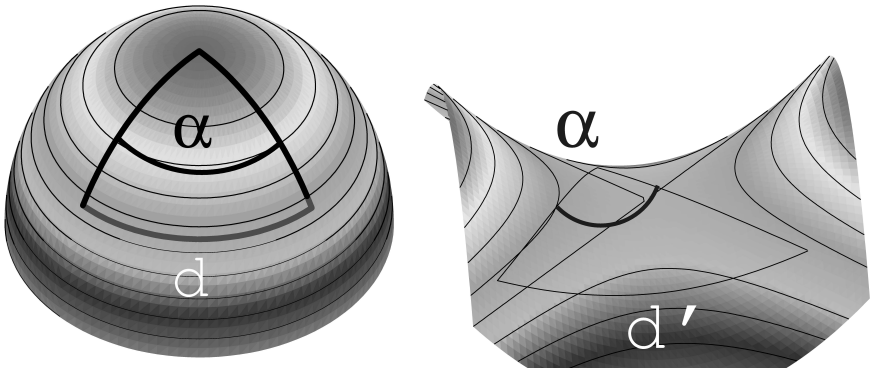
VIII. OTHER METHODS

This section is dedicated to those methods that do not belong clearly to any of the previous classes.

A. Clustering and curvature

This is one of the first attempts at detecting thematic and functional communities based on clustering [13]. Eckmann and Moses use the concept of *curvature* of a node and relate it to clustering. Consider a node i; its neighbours will be separated by a geodesic distance of at most 2. If links exist between neighbours of node i, this distance is unity. The average

distance between neighbours of any node, therefore, lies between 1 and 2. This value is directly related to clustering (see [13]). If one assumes that the distance from node i to any of its neighbours is unity, and take the distance between any of the neighbours to be the average, one can indeed think of the node to be in "curved" space, with the amount of curvature depending on the average distance between the nodes, see Fig. 5.



(a) (b)

FIG. 5: How clustering is related to curvature according to [13]. For a node i, the shortest path distance between any of its neighbours will be either 1, if the neighbours are linked, or 2, if they are not. The average distance between the neighbours can give a measure of curvature. Positive curvature is depicted in (a) and negative curvature is depicted in (b). Both triangles have sides of length unity, and the angle between the two is the same, but the distances a different, $d < d'$.

The method is based on the intuition that high curvature region of a network will belong to the same community. The authors show that finding connected components of high curvature give a good idea of community structure. In a later effort, they go on to use the method to study communities in email dialogue [47].

B. Random walk based methods

In a set of papers, Zhou and collaborators develop a methodology for community detection based on random walks [48–50]. Apart from a method for finding communities, Zhou also presents a definition of what a community is. Also worthy of note is that the method is applicable to both directed and undirected networks.

Instead of actually performing the random walk on the network, it is possible to calcu-

late the average distance from node i to node j algebraically starting with the adjacency matrix[58]. From the information contained in the average distances *local* and *global attractors* of each node can be defined. The local attractor of node i is the closest node (smallest average distance) of its nearest neighbours, and the global attractor, the node closest to all other nodes in the network. From these two definitions, two different formal definitions of community are derived. A local community is defined as a subset of the network in question whose nodes satisfy the following three conditions:

- if node i belongs to the community, then its local attractor j also belongs to the same community.
- if node i is the local attractor of any other node k, k also belongs to the same community.
- any subset of the community in question is not a local community in its own right, ie. it is the smallest possible set.

The "global community" is defined in much the same way, although in this case local communities can form part of global communities.

Apart from the definition of community this method permits a formal definition of a central node for a community. So, that node which is its own global attractor is the central node.

In a more refined effort, the author uses the average distance measure to define a *dissimilarity index* of any two nodes[59] [49]. Using the dissimilarity index, the author describes an elaborate method of hierarchical agglomeration of nodes into communities.

Most recently Zhou and Lipowsky [14, 50] present another method based on *biased random walks*. Instead of having the walkers performing purely random walks, the walker has a higher probability to jump from a node i to a node which shares the highest number of neighbours with i (essentially biasing the random walker to go down the link with the highest link clustering). This time Zhou presents an algorithm to detect communities similar to hierarchical clustering algorithms described

In a similar approach Latapy and Pons [51] also employ the intuitive idea that a random walker will get trapped for a longer time in a a densely connected community. They calculate a distance measure between two nodes, and apply an agglomerative method [52], starting

with all nodes in their own community, and joining them two by two. The main difference between this approach and the above is that at each step, the distances are recalculated. In principle this should make the method more accurate, but in reality they have very similar sensitivity, see Sec. IX.

C. Q-potts model

Another interesting approach [53] detects communities using a modified q-state Potts model[60] [54]. Here, each node is assigned a spin state between 1 and q, at random. The system is allowed to evolve using a simple Monte-Carlo method with simmulated annealing, in which the temperature is reduced incrementally. At each step, the spin of the node selected is updated using a modified q-state Potts Hamiltonian. This balances the diversity and homogeneity of the spins of the nodes, forcing them into communities. In the ground state of the system, communities are identified as groups with equal spin values, see Fig 6.

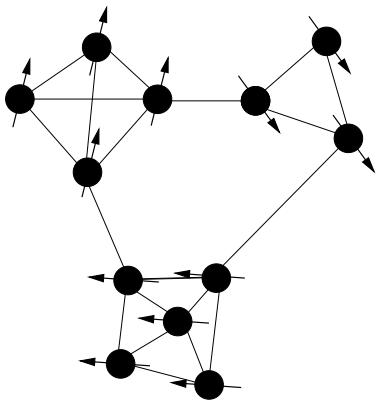


FIG. 6: The q-Potts model as applied to a small network with communities. Each node is assigned a one of q spins. As the Hamiltonian of the system is minimised, the spins in a tightly connected community take equal values, which are different to those of spins in other communities.

One of the main advantages of this algorithm is that it allows us to find fuzzy communities, since the algorithm is non deterministic neither hierarchical. The method is fast since one only needs local information to calculate the Hamiltonian. The system has two parameters, q and γ.The value of q needs to be large enough to identify all the possible communities, and γ relates the Hamiltonian with the topology of the network. Appropriate values are discussed further in the paper.

Reference	Algorithm Authors	Alias	Order
[25]	Newman, Girvan	NG	$O(m^2n)$
[10]	Girvan, Newman	GN	$O(n^2m)$
[29]	Fortunato, Latora, Marchiori	FLM	$O(n^4)$
[23]	Radicchi, Castellano, Cecconi, Loreto, Parisi	RCCLP	$O(n^2)$
[35]	Newman (Fast)	NF	$O(n\log^2 n)$
[43],	Donetti, Muñoz (Single Angular)	DMSA	$O(n^3)$
[43],	Donetti, Muñoz (Complete Angular)	DMSA	$O(n^3)$
[13]	Eckmann, Moses	EM	$O(m\langle k\rangle^2)$
[14]	Zhou, Lipowsky	ZL	$O(n^3)$
[53]	Reichardt, Bornholdt	RB	unknown
[34]	Bagrow, Bollt	BB	$O(n^3)$
[38]	Duch, Arenas	DA	$O(n^2\log n)$
[12]	Capocci, Servedio, Caldarelli, Colaiori	CSCC	$O(n^2)$
[46]	Wu, Hubberman	WH	$O(n+m)$

TABLE I: Table summarising how different approaches scale with number of nodes n and number of links m and $\langle k\rangle$ is the average degree of the network. The alias show here is used in Figures 7 and 8

IX. COMPARATIVE EVALUATION

In order for the reader to be able to compare the algorithms, both in terms of their speed and sensitivity, we would like to present a qualitative comparison for all the above community identification methods. This unfortunately is not possible for all the methods described, as they are very varied, both conceptually and in their applications.

For some of these methods we are able to estimate how the computational cost scales with network size n. Table I shows these values.

One method that has been employed in many cases is to see how the method performs when applied to ad. hoc. networks with well known, fixed community structure [25].

The networks are generated with $n = 128$ nodes, split into four communities containing 32 nodes each. A pair of nodes both of which belong to the same community are linked with

probablity p_{in}, and a pair which belong to different communities are joined with probability p_{out}. The value of p_{out} is chosen so that the average number of links a node has to members of other community, z_{out} is well known. The value of p_{in} is chosen to keep the total average node degree, z_{tot} constant. As z_{out} is increased from zero, the communities become more and more diffuse and harder to identify. Since the "real" community structure is well known in this case, it is possible to measure the number of nodes correctly classified by the method of community identification. The benchmark test, then,is to plot the fraction of correctly identified nodes as a function of z_{out}.

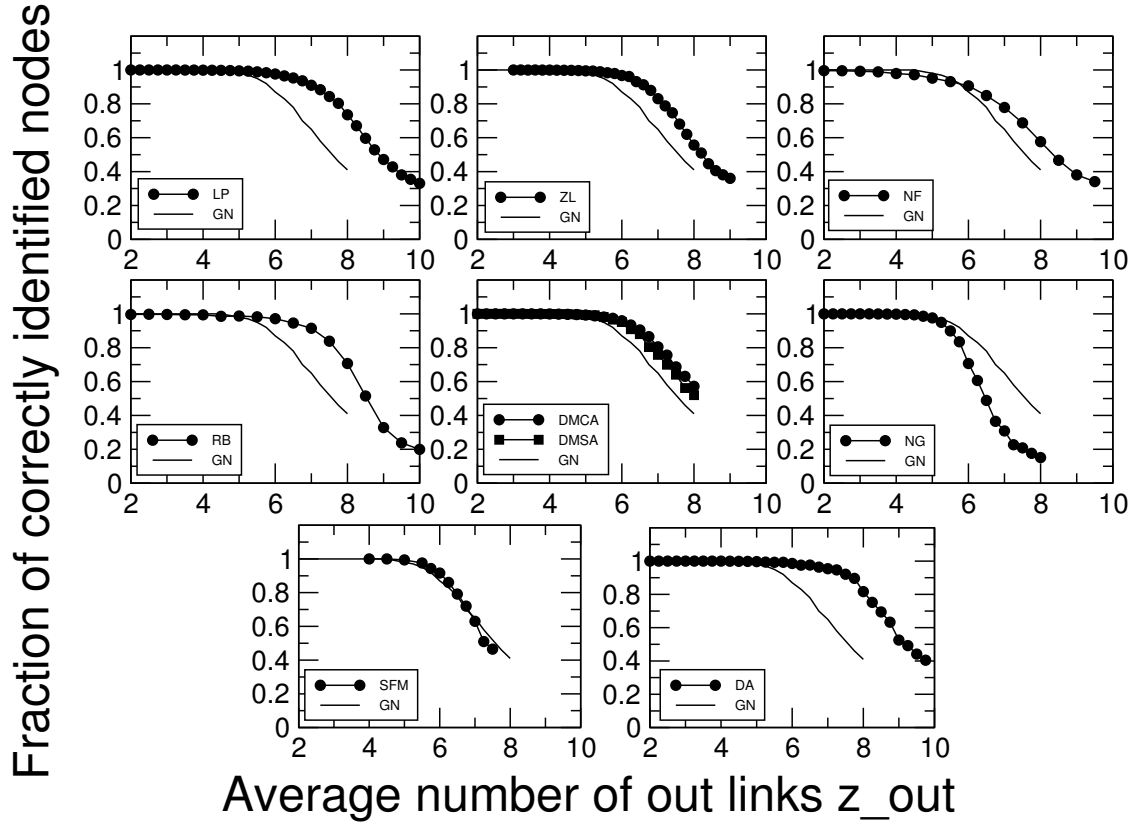


FIG. 7: Comparing algorithm sensitivity using ad. hoc. networks with predetermined community structure with $n = 128$, the network divided into four communities with 32 nodes each and total average degree of 16. The x-axis is the average number of connections to outside communities z_{out} and the y-axis is the fraction of nodes correctly identified by the method.

In Figure 7 we show the sensitivity of all methods we have been able to get gather. To summarise the large amount of information, in Figure 8 we plot the the fraction of correctly identified nodes for only three values of z_{out} (6, 7 and 8), for each method. From this we

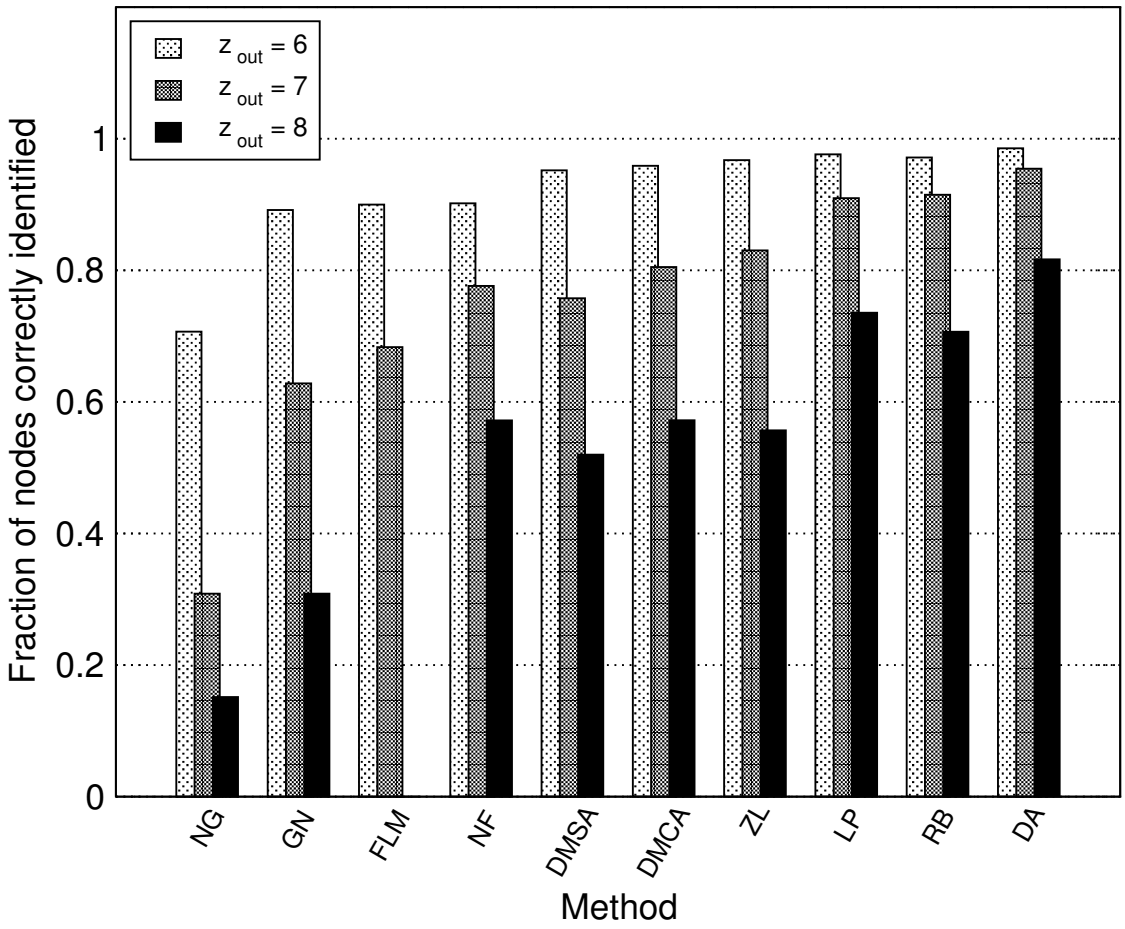


FIG. 8: The fraction of correctly identified nodes at three specific values of z_{out}, 6, 7 and 8 for all available methods. Here we can see that most of the methods are very good at finding the "correct" community structure for values of z_{out} up to 6. At $z_{out} = 7$ some methods begin to falter but most still identify more than half of the nodes correctly. At $z_{out} = 8$, only three methods are still able to identify the correct structure.

can see that most of the methods perform very well for $z_out = 6$ and even for $z_{out} = 7$ most can identify more than half the nodes correctly. For $z_{out} = 8$ there remain three methods able to identify more than half of the nodes correctly[61].

X. CONCLUSION

In this work we have attempted to give an overview of the modern approaches to community identification in complex networks. A large amount of knowledge has been collected in the field, and real progress has been made, both in the identification of communities and

their characterisation. Some questions do remain open, and it is these that we would suggest for further study: SUGERENCIAS POR FAVOR....

The authors are grateful to Luca Donetti, Haijun Zhou, Mark Newman, Santo Fortunato, Jörg Reichardt, Claudio Castellano, Matthieu Latapy and Jean-Pierre Eckmann for providing their data. LD gratefully acknowledges the funding of Generalitat de Catalunya and the COSIN project.

---

[1] A. L. Barabasi and R. Albert, Rev. Mod. Phys. **74**, 47 (2002).

[2] M. E. J. Newman, SIAM Review **45**, 167 (2003).

[3] S. N. Dorogovtsev and J. F. F. Mendes, *Evolution of Networks: From biological nets to the internet and WWW* (Oxford University Press, Oxford, 2003).

[4] R. Pastor-Satorras, M. Rubi, and A. Díaz-Guilera, eds., *Proceedings of the Conference "Statistical Mechanics of Complex Networks"* (Springer, 2003).

[5] S. Bornholdt and H. G. Schuster, eds., *Handbook of Graphs and Networks - From the Genome to the Internet* (Wiley-VCH, Berlin, 2002).

[6] M. E. J. Newman, Eur. Phys. J. B **38**, 321 (2004).

[7] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabsi, Science **297**, 1551 (2002).

[8] P. Holme, M. Huss, and H. Jeong, Bioinformatics **19**, 532 (2003).

[9] M. Boss, H. Elsinger, M. Summer, and S. Thurner, cond-mat/0309582 (2003).

[10] M. Girvan and M. E. J. Newman, Publ. Nat. Acad. Sci. **99**, 7821 (2002).

[11] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee, IEEE Computer **35**, 66 (2002).

[12] A. Capocci, V. Servedio, G. Caldarelli, and F. Colaiori, cond-mat/0402499 (2004).

[13] J.-P. Eckmann and E. Moses, Publ. Nat. Acad. Sci. **99**, 5825 (2002).

[14] H. Zhou and R. Lipowsky, preprint (2005).

[15] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness* (W. H Freeman, New York, 1979).

[16] B. W. Kernighan and S. Lin, The Bell System Tech J **49**, 291 (1970).

[17] M. Fiedler, Czech. Math. J. **23**, 298 (1973).

[18] S. Boettcher and A. G. Percus, Phys Rev E **64** (2001).

[19] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas, Phys. Rev. E **68** (2003).

[20] P. Gleiser and L. Danon, Advances in Complex Systems **6**, 565 (2003).

[21] A. Arenas, L. Danon, A. Diaz-Guilera, P. M. Gleiser, and R. Guimerà, Eur. Phys. J. B **38**, 373 (2004).

[22] S. Wasserman and K. Faust, *Social Network Analysis, Methods and Applications* (Cambridge University Press, 1994).

[23] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, Publ. Nat. Acad. Sci. **101**, 2658 (2004).

[24] C. Bron and J. Kerbosch, Communications of the ACM pp. 575–577 (1973).

[25] M. E. J. Newman and M. Girvan, Phys. Rev. E **69**, 026113 (2004).

[26] R. Guimerà, M. Sales, and L. N. A. Amaral, Phys. Rev. E **70**, 025101 (2004).

[27] M. E. J. Newman, Phys. Rev. E **64**, 016132 (2001).

[28] U. Brandes, Journal of Mathematical Sociology **25**, 163 (2001).

[29] S. Fortunato, V. Latora, and M. Marchiori, Phys. Rev. E **70** (2004).

[30] V. Latora and M. Marchiori, cond-mat/0402050–to appear in Elsevier?? (2004).

[31] J. Scott, *Social Network Analysis, a handboook* (SAGE publications, 2000).

[32] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data* (Prentice-Hall, Upper Saddle River, NJ, 1988).

[33] J. Hopcroft, O. Khan, B. Kulis, and B. Selman, Publications of the National Academy of Sciences **101**, 5249 (2004).

[34] J. P. Bagrow and E. M. Bollt, cond-mat/0412482 (2004).

[35] M. E. J. Newman, Phys. Rev. E **69** (2004).

[36] A. Clauset, M. E. J. Newman, and C. Moore, Phys. Rev. E **70**, 06111 (2004).

[37] C. P. Massen and J. P. K. Doye, cond-mat/0412469 (2004).

[38] J. Duch and A. Arenas, cond-mat/0501368 (2005).

[39] S. Boettcher and A. G. Percus, Phys. Rev. Lett. **86**, 5211 (2001).

[40] B. Bollobas, *Modern Graph Theory* (Springer, New York, 1998).

[41] A. Pothen, SIAM Journal on Matrix Analysis and Applications **11**, 430 (1990).

[42] A. Pothen, *Parallel Numerical Algorithms* (Kluwer Academic Press, 1996), chap. Graph partitioning algorithms with applications to scientific computing.

[43] L. Donetti and M. A. M. noz, J. Stat. Mech.: Theor. Exp. (2004).

[44] G. H. Golub and C. F. V. Loan, *Matrix Computations* (Johns Hopkins University Press, Baltimore, 1996).

[45] M. Steyvers and J. B. Tenenbaum (2001), cond-mat/0110012.

[46] F. Wu and B. Huberman, Eur. Phys. J. B **38**, 331 (2004).

[47] J.-P. Eckmann, E. Moses, and D. Sergi, Publications of the National Academy of Sciences **101**, 14333 (2004).

[48] H. Zhou, Phys. Rev. E **67**, 041908 (2003).

[49] H. Zhou, Phys. Rev. E **67**, 061901 (2003).

[50] H. Zhou and R. Lipowsky, Lecture Notes in Computer Sciences (in press) (2004).

[51] M. Latapy and P. Pons, cond-mat/0412568 (2004).

[52] J. H. Ward, Journal of the American Statistical Association **53**, 263 (1063).

[53] J. Reichardt and S. Bornholdt, Phys. Rev. Lett. **93**, 218701 (2004).

[54] M. Blatt, S. Wiseman, and E. Domany, Physical Review Letters **76**, 3251 (1996).

[55] The membership matrix is not necessarily symmetric: an L-shell starting at node i may class node j as being in the same community, but this does not imply that an L-shell starting at j will class i as being in the same community.

[56] This is the Metropolis criterion.

[57] The name Laplacian is drawn from the fact that applying the discrete Laplacian operator on the network in question gives the Laplacian matrix.

[58] Note that although the absolute distance from a node to one of its neighbours is necessarily 1, this is not generally true for a random walker, which could easily make a "mistake" and go where it is not supposed to.

[59] For nodes i and j the dissimilarity index is simply the square of the difference between the distance from another node k to i and the distance from k to j summed over all nodes k.

[60] The q-Potts model is essntially an Ising model with q states instead of just two.

[61] At this point is important to note that there may be some differences in the way the authors calculated these values.