Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 502809, 15 pages
http://dx.doi.org/10.1155/2014/502809

Research Article

Performance Evaluation of Modularity Based Community Detection Algorithms in Large Scale Networks

Vinícius da Fonseca Vieira,[1,2] Carolina Ribeiro Xavier,[1,2] Nelson Francisco Favilla Ebecken,[2] and Alexandre Gonçalves Evsukoff[2]

[1]*Department of Computer Science, Federal University of São João del Rei (UFSJ), 36301-360 São João del Rei, MG, Brazil*
[2]*COPPE, Federal University of Rio de Janeiro (UFRJ), P.O. Box 68506, 21941-972 Rio de Janeiro, RJ, Brazil*

Correspondence should be addressed to Vinícius da Fonseca Vieira; vinicius@ufsj.edu.br

Received 28 August 2014; Accepted 27 November 2014; Published 28 December 2014

Academic Editor: Mohamed A. Seddeek

Community structure detection is one of the major research areas of network science and it is particularly useful for large real networks applications. This work presents a deep study of the most discussed algorithms for community detection based on modularity measure: Newman's spectral method using a fine-tuning stage and the method of Clauset, Newman, and Moore (CNM) with its variants. The computational complexity of the algorithms is analysed for the development of a high performance code to accelerate the execution of these algorithms without compromising the quality of the results, according to the modularity measure. The implemented code allows the generation of partitions with modularity values consistent with the literature and it overcomes 1 million nodes with Newman's spectral method. The code was applied to a wide range of real networks and the performances of the algorithms are evaluated.

1. Introduction

Community detection is of great interest in the field of complex networks and its study has been subject of many works [1–6]. A consensual notion about the characterization of a community in a network is a subset of nodes with great internal density and low external density.

Several works can be found in the literature analyzing and comparing different measures for quality of partitions. For instance, in the work of Yang and Leskovec [7], the authors investigate the suitability of several measures to characterize ground-truth based communities. The work of Moradi et al. [8] compares different quality functions regarding their ability to classify useful and spam messages in an email network.

Currently, modularity, proposed by Newman and Girvan [9], is the most widely adopted measure for the assessment of the quality of communities in networks. To a particular community, modularity can be understood, in a general way, as the difference between the fraction of edges inside the community and the fraction of edges expected by a random version of the network, preserving the degree distribution of the nodes.

In one of the first works with the purpose of investigating community structures in networks, Girvan and Newman [10] propose a method based on edge centrality [11], able to handle small-scale networks (up to 1000 nodes). Later, Newman proposes a modularity based heuristic method, able to handle networks on a larger scale (up to hundreds of thousands of nodes) [12]. In order to adapt the heuristic method to large scale networks, Clauset et al. define, thus, a methodology which allows it to be executed more efficiently [1].

In a different direction, some works use approximate optimization methods for community detection in complex networks based on the modularity measure [13, 14], such as Newman's work [15], which proposes a relaxed optimization method based on spectral graph theory [16].

Recently, there has been great discussion about some negative aspects in the use of modularity as a measure of the quality of the division of a network in communities. The

work of Fortunato and Barthélemy is worth mentioning [17], which verifies that the modularity can fail in the identification of intuitive communities (for instance, cliques of nodes). This problem is broadly addressed in the literature [4, 18–20] and frequently related as the resolution limit problem. Another aspect which is important to point out in the use of modularity is the fact that some communities can show high modularity values, even with just few variations from random connections, as observed by Guimerà et al. [21] and also discussed by Kehagias [22]. Furthermore, one must consider the fact that very distinct partitions can lead to similar modularity values, as discussed by Good et al. [18], which is a drawback in the use of modularity.

The discussion concerning the use of the modularity as a measure of the quality of partitions, introduced by the aspects previously mentioned, is of primary importance in the community detection area. Nevertheless, several works as [4, 18, 20] discuss that modularity is still a very appropriate measure for the assessment of community structures in networks and emphasize the importance of the investigation on methods for modularity maximization.

Currently, the growing possibility of storing and processing data in high performance computing environments raises the demand for the analysis of increasingly larger networks, with million (or even billion) nodes. A great challenge set in the complex networks area is the identification of communities in large scale networks.

There is a great demand for computational methods that are capable of detecting community structure in large scale networks and this is currently one of the most important problems in the area of complex networks. Several works can be found in the literature with the purpose of proposing and studying such methods and, among them, [1, 2, 4, 9, 15, 23–27] can be cited.

The method of Clauset, Newman, and Moore (CNM) can be considered one of the most important methods for community detection in networks and, currently, it is one of the most studied methods with this purpose. Some modifications can be found in the literature in order to accelerate its execution and make it possible to investigate larger networks, including the works of Wakita and Tsurumi [28], Leon-Suematsu and Yuta [4], and Danon et al. [24]. Another important heuristic method for community detection in large scale networks is reported in the work of Blondel et al. [29], which uses an agglomerative multistep process during its execution. Currently, there has been a great interest in nonparametric methods, which aim at adjusting networks to statistic models, according to its structural properties.

This work aims at the investigation of the computational issues of methods for community detection which enable them to deal with large scale networks. Two of the most adopted modularity based methods for community detection are addressed: the spectral method of Newman [15] combined with a variation of the Kernighan-Lin method [30], which is called fine-tuning, and the method of Clauset, Newman, and Moore (CNM) [1]. Some variations on the fine-tuning stage are also proposed in order to accelerate its execution without harming the quality of the result obtained.

The computational implementation of the studied methods is discussed in respect of the computational complexity of the algorithms. The implemented algorithms are used to qualitative and quantitative comparative study of the spectral method of Newman and the CNM method, adjusting their application to large scale networks. All of the developed code is freely available for download on the web, in Github repository (http://www.github.com/vfvieira/).

The remainder of the work is organized as follows. Section 2 presents the problem of community detection and the methods addressed in this work. In Section 3, the main computational issues concerning the implementation of the methods are presented. The experiments performed, as well as the obtained results and discussion, are presented in Section 4. Section 5 presents some conclusions and future works.

2. Community Detection in Networks

2.1. Problem Statement. A community structure in a network can be identified when there is a division of the network in groups with high density of internal connections and, at the same time, low density of external connections. The community sense becomes more evident as the difference between the intragroup and intergroup increases. Thus, it is a central concern to quantify the quality of a particular division of the network in communities.

Consider a graph $G(\mathbb{V}, \mathbb{E})$ where $\mathbb{V}$ represents the set of n nodes and $\mathbb{E}$ represents the set of m edges. In this work, the edges of the graph are unweighted and undirected. Thus, the graph G is represented by an adjacency matrix $\mathbf{A}$, where an element $\mathbf{A}_{ij} = 1$, if a node v_i is connected to a node v_j and $\mathbf{A}_{ij} = 0$, otherwise.

A community structure $\mathbb{C}$ is defined as a partition of $\mathbb{V}$ in nc communities:

$$\mathbb{C} = \{\mathbb{C}_i,\ i = 1, \ldots, nc\}, \tag{1}$$

where each community $\mathbb{C}_i \subset \mathbb{V}$ is a subset of nodes of G such as

$$\mathbb{V} = \bigcup_{i=1}^{nc} \mathbb{C}_i, \qquad \bigcap_{i=1}^{nc} \mathbb{C}_i = \emptyset. \tag{2}$$

Equations (2) determine that a community structure defines a partition of the set of nodes, such that there is no overlap between the communities. Alternatively, several approaches that consider the overlapping of communities can be found in the literature [7, 31].

The quality of a community structure can be assessed by modularity, a measure proposed by Newman and Girvan [6], which considers the difference between the fraction of edges in a community and the fraction of edges expected by a network with the same degree distribution, but randomly placed.

Consider m as the number of edges in the network and $\mathbf{k}$ as the degree vector, where $\mathbf{k}_i$ is the degree of a node v_i. The

number of connections expected between all pairs of nodes inside the same community is $(1/2)\sum_{ij}(\mathbf{k}_i\mathbf{k}_j/2m)\delta(c_i, c_j)$, where $c_i = a$ denotes that the node v_i belongs to the community $\mathbb{C}_a$ and $\delta(\cdot,\cdot)$ is the Kronecker delta, which returns 1 if the operands are equal and 0 otherwise. The factor $1/2$ is used to avoid double counting of edges.

Modularity Q of a community structure can be defined as

$$Q = \frac{1}{2m}\sum_{ij}\left(\mathbf{A}_{ij} - \frac{\mathbf{k}_i\mathbf{k}_j}{2m}\right)\delta\left(c_i, c_j\right). \tag{3}$$

From the definition of modularity, it can be said that, for a particular network, a community structure that corresponds to the maximum value of modularity is the best partition of the set of nodes. Based on this principle, one has an important motivation for the modularity maximization for solving the community detection problem in complex networks. Modularity optimization in networks has been subject of several works in the literature, including [4, 15, 24, 27–29, 32, 33].

This work focuses on two of them: the spectral method of Newman and the heuristic method of Clauset, Newman, and Moore. The next sections are dedicated to such methods.

2.2. Spectral Optimization of Modularity. The spectral approach was applied by Newman and Girvan to the community detection problem [6] and, to this end, they define a modularity matrix $\mathbf{B}$, in which each element $\mathbf{B}_{ij}$ can be defined as

$$\mathbf{B}_{ij} = \mathbf{A}_{ij} - \frac{\mathbf{k}_i\mathbf{k}_j}{2m}. \tag{4}$$

Considering the division of the network in just two generic communities $\mathbb{C} = \{\mathbb{C}_1, \mathbb{C}_2\}$, the communities can be represented such that each node belongs to a vector $\mathbf{s} \in \{-1, 1\}^n$, and $\mathbf{s}_i = +1$ if $v_i \in \mathbb{C}_1$ and $\mathbf{s}_i = -1$ if $v_i \in \mathbb{C}_2$, redefining modularity Q (3) in terms of $\mathbf{B}$ as

$$Q = \frac{1}{4m}\mathbf{s}^T\mathbf{B}\mathbf{s}. \tag{5}$$

Relaxing the vector $\mathbf{s}$ in a vector $\mathbf{u}$ which allows any real number, the solution of the modularity maximization problem can be obtained by solving the eigenproblem

$$\mathbf{B}\mathbf{u}_\mathbf{1} = \lambda_1\mathbf{u}_\mathbf{1}, \tag{6}$$

where λ_1 is the largest eigenvalue of $\mathbf{B}$ and $\mathbf{u}_\mathbf{1}$ is its corresponding eigenvector. For the sake of simplicity, λ_1 and $\mathbf{u}_\mathbf{1}$ will be treated, respectively, as λ and $\mathbf{u}$ for the remainder of the work.

The solution of (6) maximizes the approximation $Q = \mathbf{u}^T\mathbf{B}\mathbf{u}$. From that, the community structure is defined by the eigenvector corresponding to the largest eigenvalue of $\mathbf{B}$, according to the signal of $\mathbf{u}$: the nodes corresponding to the positive elements of $\mathbf{u}$ are assigned to a group and the nodes corresponding to the negative elements of $\mathbf{u}$ are assigned to the other group, which can be better described as

$$\widehat{\mathbf{s}}_i = \begin{cases} +1, & \text{if } u_i \geqslant 0 \\ -1, & \text{if } u_i < 0, \end{cases} \qquad i = 1, \ldots, n. \tag{7}$$

This method is known in the literature as Newman's bisection method, which aims at dividing a network into two communities (generically defined as $\mathbb{C}_1$ and $\mathbb{C}_2$), and can be summarized by the following steps: calculate the eigenvector corresponding to the largest eigenvalue of the modularity matrix; assign the nodes to the communities according to the sign of the elements (positive elements are assigned to a community $\mathbb{C}_1$ and negative elements are assigned to the other community $\mathbb{C}_2$).

In order to generalize the method for the division of the network in several communities, the maximization of modularity Q can be performed in a successive bisection process. Thus, the method evaluates if there is a gain in the modularity obtained from the division of the network (or a community), and if convenient, a division of the nodes into two subsets is done. In a recursive process, the method evaluates if it is convenient to divide each of the two subsets, and if the division increases the modularity, the operation is performed. The process stops when there is no division in which the modularity will be increased.

However, the strategy of simply removing the vertices which connect two communities and applying the method to each community leads to an essential mistake in definition of the modularity. As defined by (3), modularity to be maximized must consider the whole network.

In this sense, Newman defines a community modularity matrix $\mathbf{B}^{(\mathbb{C}_a)}$, which concerns only a particular community, $\mathbb{C}_a$ in this case, and can be defined as

$$\mathbf{B}_{ij}^{(\mathbb{C}_a)} = \mathbf{B}_{ij} - \delta\left(i, j\right)\sum_{v_k \in \mathbb{C}_a}\mathbf{B}_{ik}. \tag{8}$$

Then, Newman defines a measure ΔQ which evaluates the modularity variation caused by the division of a generic community $\mathbb{C}_a$ and can be written as

$$\Delta Q = \frac{1}{4m}\mathbf{s}^T\mathbf{B}^{(\mathbb{C}_a)}\mathbf{s}. \tag{9}$$

This definition allows the method to be applied to any generic community, since the sum of the rows of $\mathbf{B}^{(\mathbb{C}_a)}$ is still zero and ΔQ is also zero when the community remains undivided.

In summary, generalized Newman's method works as follows: the communities are repeatedly divided according to the signs of the leading eigenvector of its corresponding modularity matrix; when a division does not result in a positive change in ΔQ for a community, it must remain undivided; when there is no community in which the division increases ΔQ, the process is finished. Algorithm 1 shows an algorithm for generalized Newman's method.

Input: A network $G = (\mathbb{V}, \mathbb{E})$
Output: A community structure $\mathbb{C} = \{\mathbb{C}_a,\ a = 1, \ldots, nc\}$
(1) $\mathbb{C} \leftarrow \{\varnothing\}$;
(2) $Q \leftarrow 0$;
(3) $nc \leftarrow 1$;
(4) $\mathbb{C}' \leftarrow \mathbb{V}$
(5) *NewmanSpectral*($\mathbb{C}'$);
(6) **forall the** *NewmanSpectral* **do**
(7) Calculate the modularity matrix **B** for $\mathbb{C}'$;
(8) Find the eigenvector **u** related to the largest algebraic eigenvalue of **B** (6);
`// Will split community` $\mathbb{C}'$ `in` $\mathbb{C}''$ `and` $\mathbb{C}'''$;
(9) $\mathbb{C}'' \leftarrow \{\varnothing\}$;
(10) $\mathbb{C}''' \leftarrow \{\varnothing\}$;
(11) **foreach** element i of **u** **do**
(12) **if** $u_i \geqslant 0$ **then**
(13) $\mathbf{s}_i \leftarrow +1$;
(14) $\mathbb{C}'' \leftarrow \mathbb{C}'' \cup v_i$
(15) **else**
(16) $\mathbf{s}_i \leftarrow -1$;
(17) $\mathbb{C}''' \leftarrow \mathbb{C}''' \cup v_i$
(18) **end**
(19) **end**
(20) Calculate the modularity variation ΔQ with vector **s** (9);
(21) **if** $\Delta Q \geqslant 0$ **then**
`// Will try to split communities` $\mathbb{C}''$ `and` $\mathbb{C}'''$;
(22) $Q \leftarrow Q + \Delta Q$;
(23) *NewmanSpectral*($\mathbb{C}''$);
(24) *NewmanSpectral*($\mathbb{C}'''$);
(25) **else** `// Found a community` $\mathbb{C}'$
(26) $\mathbb{C}_{nc} \leftarrow \mathbb{C}'$;
(27) $\mathbb{C} \leftarrow \mathbb{C}\, U\, \mathbb{C}_{nc}$;
(28) $nc \leftarrow nc + 1$;
(29) **end**
(30) **end**

Algorithm 1: Newman's spectral method for several communities.

2.2.1. Kernighan-Lin Method for Community Detection. Even though Newman's spectral method leads to high quality communities, it can be substantially improved when post-processing to each bisection is performed, as suggested by Newman [15]. Newman proposes a variation in the method of Kernighan-Lin, a well known graph partitioning method, which allows it to be applied to the communities bisection problem, which is called fine-tuning stage. The main importance of Kenrnighan-Lin method does not lie in the method itself but in its combination to Newman's spectral method.

The original formulation of Kernighan-Lin method [34] differs from the method for community detection in some aspects. In the graph partitioning problem, the method aims at the minimization of the cut size and a constraint on the size of each group may be respected. In the community detection problem, the quantity to be optimized is modularity, which must be maximized, and the sizes of the communities are unconstrained.

Kernighan-Lin method is based on a very intuitive notion of division of nodes in groups and works with an initial division of the node set in two generic subsets $\mathbb{C}_a$ and $\mathbb{C}_b$, which can be performed randomly or in a convenient way (in the case of the method used as fine-tuning for Newman's spectral method, the initialization of the groups is performed by the division obtained as the solution of (6)). Among all the nodes, the method finds which node that, when moved (from $\mathbb{C}_a$ to $\mathbb{C}_b$ or from $\mathbb{C}_b$ to $\mathbb{C}_a$), causes the largest increase (or the least decrease) to the modularity. These operations are performed repeatedly, with the constraint that each node can be moved only once. The method finds, among all the intermediate states, the one which leads to the largest modularity value.

Algorithm 2 shows the steps for the execution of Kernighan-Lin method adapted to community detection.

2.3. Method of Clauset, Newman, and Moore (CNM). The method of Clauset, Newman, and Moore (CNM) [1] is a heuristic method aiming at the fast identification of communities, suited for large scale networks. CNM is one of the most cited methods in the literature which focuses on dealing with large networks. As CNM is a greedy heuristic method, its application may lead to partitions which differs from the optimal solution, and, in many cases, modularity obtained is much lesser than the values found by other methods.

Input: Network $G = (\mathbb{V}, \mathbb{E})$;
Two communities $\mathbb{C}''$ and $\mathbb{C}'''$;
A division vector **s** (7)
Output: Two communities $\mathbb{C}''$ e $\mathbb{C}'''$;
A division vector **s** (after post-processing)
(1) $\Delta Q \leftarrow 0$;
(2) *count* $\leftarrow 0$;
(3) $\mathbf{s}^* \leftarrow \mathbf{s}$;
(4) **while** *count* $< n$ **do**
(5) $\delta Q^* \leftarrow -1$;
(6) δQ^* *_id* $\leftarrow 0$;
(7) element i of vector **s** **if** *element i not moved yet* **then**
(8) Evaluates gain δQ when moving i from $\mathbb{C}''$ to $\mathbb{C}'''$ (or from $\mathbb{C}'''$ to $\mathbb{C}''$);
(9) **if** $\delta Q > \delta Q^*$ **then**
(10) $\delta Q^* \leftarrow \delta Q^*$;
(11) δQ^* *_id* $\leftarrow i$;
(12) **end**
(13) **end**
(14) Update vector **s** moving δQ^* *_id* from $\mathbb{C}''$ to C''' or from $\mathbb{C}'''$ to $\mathbb{C}''$;
(15) *count* $\leftarrow$ *count* $+ 1$;
(16) Assign element i as moved;
(17) **if** $\Delta Q + \delta Q > \Delta Q$ **then**
(18) $\mathbf{s}^* \leftarrow \mathbf{s}$;
(19) $\Delta Q \leftarrow \Delta Q + \delta Q$;
(20) **end**
(21) **end**

Algorithm 2: Kernighan-Lin method adapted to community detection.

The algorithm, proposed by Newman [12], according to its original description (later improved by Clauset et al. [1]), initially associates each node of the network with a community. Then, it repeatedly combines the communities of which the union produces the highest increase to the modularity Q of the community structure.

The method aims at finding the combination of communities that results in the largest increase in Q and then performs such operation. That is, the method finds the pair of communities $\mathbb{C}_a$ and $\mathbb{C}_b$ that leads to the highest modularity value when combined. Such value can be interpreted as an affinity measure between two generic communities $\mathbb{C}_a$ and $\mathbb{C}_b$; thus it aims at finding two similar communities so that they can be joined [24].

CNM method uses an agglomerative strategy and, considering a network with n nodes (and consequently n unary initial communities), after $n - 1$ combinations the result is only one community containing all the nodes, and the algorithm stops.

Newman [12] proposes a strategy to evaluate the gain obtained by the union of two generic communities $\mathbb{C}_a$ and $\mathbb{C}_b$, taking an $n \times n$ matrix as basis (which, initially, will be identical to the adjacency matrix). The union of two communities $\mathbb{C}_a$ and $\mathbb{C}_b$ corresponds to the substitution of the a^{-th} and b^{-th} lines (and columns) of the matrix by their sums. However, as shown by Clauset et al. [1], the sparsity of the matrix results in a memory waste and a high execution cost to apply the union of communities over the complete lines/columns.

Thus, Clauset et al. propose a matrix **M** in order to store the modularity gain caused by the union of two generic communities $\mathbb{C}_a$ and $\mathbb{C}_b$, keeping just the elements $\mathbf{M}_{ab}$ linked by at least one edge [1]. That is, **M** only stores the elements $\mathbf{M}_{ab}$ (the modularity gain obtained by the union of $\mathbb{C}_a$ and $\mathbb{C}_b$) when the communities are connected. The elements $\mathbf{M}_{ab}$ of **M** are initialized as

$$M_{ab} = \begin{cases} \dfrac{1}{2m} - \dfrac{\mathbf{d}_a \mathbf{d}_b}{(2m)^2}, & \text{if } \mathbb{C}_a \text{ and } \mathbb{C}_b \text{ are connected} \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

where **d** is a vector which stores the sum of the degrees of the nodes belonging to a community $\mathbb{C}_a$ and the elements $\mathbf{d}_a$ are defined as

$$\mathbf{d}_a = \sum_i \mathbf{k}_i, \quad v_i \in \mathbb{C}_a. \tag{11}$$

After calculating the initial value of **M**, the method performs successive unions of communities (updating the matrix **M** for each union), until no more gain in the modularity can be obtained. For the union of a particular pair of communities $\mathbb{C}_a$ and $\mathbb{C}_b$ (where the resulting community is stored as $\mathbb{C}_a$), only the line and the column indexed by a must be updated. In addition, the line and the column indexed by b must be removed, since community $\mathbb{C}_b$ no longer exists. Thus, Clauset et al. define a set of rules to update the whole matrix **M** with

Input: A network $G = (\mathbb{V}, \mathbb{E})$
Output: A community structure $\mathbb{C} = \{\mathbb{C}_a,\ a = 1, \ldots, nc\}$
(1) Calculate the initial values for $\mathbf{M}$ (10);
(2) Calculate the initial modularity value Q;
(3) $nc \leftarrow n$;
(4) **repeat**
(5) Join the pair of communities $\mathbb{C}_a$ and $\mathbb{C}_b$ corresponding to the highest value of $\mathbf{M}$ ($\max(\mathbf{M})$): $\mathbf{M}_{ab}$;
(6) Update matrix $\mathbf{M}$ (12);
(7) $nc \leftarrow nc - 1$;
(8) $Q = Q + \mathbf{M}_{ab}$;
(9) **until** $\max(\Delta Q) < 0$;

Algorithm 3: Algorithm of Clauset, Newman, and Moore.

respect to the connectivity of communities $\mathbb{C}_a$ and $\mathbb{C}_b$, which are being combined, to other communities $\mathbb{C}_c$ [1]:

$$\mathbf{M}'_{ac} = \begin{cases} \mathbf{M}_{ac} + \mathbf{M}_{bc}, \\ \quad \text{if } \mathbb{C}_c \text{ is connected to both } \mathbb{C}_a \text{ and } \mathbb{C}_b \\ \mathbf{M}_{bc} - 2\left(\dfrac{\mathbf{d}_a}{2m}\right)\left(\dfrac{\mathbf{d}_c}{2m}\right), \\ \quad \text{if } \mathbb{C}_c \text{ is connected to } \mathbb{C}_b \text{ but not to } \mathbb{C}_a \\ \mathbf{M}_{ac} - 2\left(\dfrac{\mathbf{d}_b}{2m}\right)\left(\dfrac{\mathbf{d}_c}{2m}\right), \\ \quad \text{if } \mathbb{C}_c \text{ is connected to } \mathbb{C}_a \text{ but not to } \mathbb{C}_b. \end{cases} \tag{12}$$

CNM method can be described by the steps presented by Algorithm 3.

2.3.1. Variation of Danon, Diaz, and Arenas (DDA). Variations on CNM can be found in the literature, some of them aiming at improving the modularity obtained and some of them aiming at the reduction of the execution time (possibly harming the quality of the partition). Among them, the variation of Danon, Diaz, and Arenas (DDA) [24] can be highlighted and is primarily motivated by the difference between the size of the communities obtained by the community detection methods.

By analysing how the sizes of communities generated by CNM affect the efficiency of the algorithm, Danon, Diaz, and Arenas noticed that some of the method features lead it to a typical behaviour. At the beginning of the execution, each community has only one node. Thus, based on (10), modularity gain is higher for the combination corresponding to the least degrees product. As these communities are joined, the resulting community will absorb a community adjacent to the previously joined communities. The growing community tends to increasingly absorb the adjacent nodes, leading the process to form a few large groups, and this happens specially in networks with a high clustering coefficient.

In order to avoid this behaviour, DDA propose a simple modification in the choice of communities $\mathbb{C}_a$ and $\mathbb{C}_b$ to be combined (line 5 of Algorithm 3), the normalization of $\mathbf{M}$ by the number of connections in the communities:

$$\mathbf{M}'^{\mathrm{DDA}}_{ab} = \frac{\mathbf{M}_{ab}}{\mathbf{d}_a}. \tag{13}$$

This modification only affects the choice of communities to be combined and not the variation of modularity obtained in a particular stage and the real number of $\mathbf{M}_{ab}$ must be calculated. It is important to notice that this measure is asymmetric; that is, $\mathbf{M}'^{\mathrm{DDA}}_{ab} \neq \mathbf{M}'^{\mathrm{DDA}}_{ba}$, which is not a problem, since both $\mathbf{M}_{ab}$ and $\mathbf{M}_{ba}$ are considered.

Danon et al. highlight that, in the earlier iterations of CNM, the original method shows higher modularity values [24]. However, as the execution proceeds, the modification of DDA tends to provide higher quality partitions.

The need to calculate the real value of $\mathbf{M}$, theoretically, can lead to an increase in the execution time. However, in practice, as it will be discussed in Section 3.3, a fewer number of unions must be performed, since the communities sizes are more balanced. This leads to a significant reduction in the execution time.

3. Computational Issues and Implementation of Community Detection Methods

This section aims at describing in detail the implementation of the method of Clauset, Newman, and Moore and Newman's spectral method performed in this work. Computational issues and relevant design decisions considered in the development of the algorithms are presented. Thus, the aspects described in this section actually cause a significant impact on the algorithms results, both from quality and execution complexity points of view.

The methods were chosen aiming at the definition of a set of computational tools which are able to deal with the modularity optimization for community detection under different approaches: divisive × agglomerative; heuristic solution × relaxed solution.

The methods were all implemented with free software, aiming at letting them in fair comparison conditions. In order to do this, as far as possible, a set of appropriate data structures and computational tools were adopted.

The implemented code is freely available in Github repository and can be accessed in http://www.github.com/vfvieira/.

3.1. Newman's Spectral Method. Newman's spectral method for community detection was implemented in ANSI C language with the scientific library PETSc (*Portable, Extensible Toolkit for Scientific Computation*) [35], which includes a set of mathematical routines and data structure for the implementation of large scale serial or parallel scientific applications. In this work, PETSc version 3.3 was used (http://www.mcs.anl.gov/petsc/).

One of the major concerns in the development of the computational tool relates to matrix data storage, since real world networks are frequently highly sparse [36]. In this work, compressed sparse row (CSR) storage scheme was adopted, the most widely used scheme for sparse matrices [36]. CSR uses three vectors to store values. Considering a matrix with n lines and nz nonzero elements, a CSR matrix uses a vector val with nz elements, which stores sequentially the nonzero elements, a vector ind with nz elements which stores the columns of each element of $\mathbf{A}$, and a vector ptr with $n + 1$ elements, which stores the index of the element in val which is the first nonzero element of each line.

Another central concern is the calculation of the dominant eigenvector performed at each bisection. In this work, power method was used, which corresponds to a sequence of matrix-vector multiplications and, according to Heath [37], it is one of the most simple and direct approaches for obtaining the dominant eigenvalue, which is enough for the scope of this work.

Regarding time complexity, a matrix-vector multiplication costs $O(n^2)$ when performed in a simple way. However, considering a sparse matrix, such as a network, the multiplication can be performed in $O(m)$, where m is the number of nonzero elements of the matrix.

In order to calculate the computational cost of the whole method, the number of iterations for the convergence of the eigenvalue in the power method must be considered. In the worst case scenario, this number can reach n, which makes the power method run in $O(mn)$. In practice, however, this value can be much lower. In other words, power method tends to run in $O(n^2)$ when executed in a dense matrix and tends to run in $O(n)$ when executed in a sparse matrix.

Thus, a major problem in the calculation of the dominant eigenvector of modularity matrix $\mathbf{B}$ arises from the fact that $\mathbf{B}$ is not a sparse matrix and, frequently, all of its elements are nonzero even when the network is sparse. To overcome this, Newman [15] proposes the multiplication $\mathbf{Bx}$, where $\mathbf{x}$ is the vector to which the dominant eigenvector will converge, to be performed on a sparse matrix. In order to do this, the operation is based on (4) (which defines the modularity matrix $\mathbf{B}$), and the multiplication $\mathbf{Bx}$ can be written as

$$\mathbf{Bx} = \mathbf{Ax} - \frac{\mathbf{k}\left(\mathbf{k}^T\mathbf{x}\right)}{2m}. \tag{14}$$

The first term $\mathbf{Ax}$ is a multiplication of the vector $\mathbf{x}$ by a sparse matrix, which can be executed in $O(m)$. The second term corresponds to an inner product, which is executed in $O(n)$. Thus, whole $\mathbf{Bx}$ multiplication is performed in $O(m+n)$. Considering that n multiplications are needed by the process to converge to the dominant eigenvalue, the total time for the eigenvector calculation is executed in $O(mn)$. Then, in the case of a sparse network, the execution time complexity is similar to $O(n^2)$.

In a case in which the network is divided into more than two parts, as in most real situations, the division is repeated until the components can no longer be divided. The execution time depends, then, on the depth of the division tree. In the worst case, this depth is linear in n; however, in a more realistic case, the depth of the tree is $\log n$. Thus, the total algorithm execution time for community detection in networks is $O(n^2 \log n)$ for a sparse network.

Another problem, which arises when the power method is applied to $\mathbf{B}$, is that Newman's method uses the eigenvector corresponding to the largest eigenvalue of $\mathbf{B}$. However, power method converges to the dominant eigenvector, that is, the eigenvector corresponding to the eigenvalue with the largest magnitude (and it is not guaranteed the the largest eigenvalue in magnitude is the largest algebraic eigenvalue).

To overcome this problem, a shift can be performed in the eigenvalue problem, using the eigenvalue with the largest module. Thus, after the application of the power method to $\mathbf{B}$, the dominant eigenvalue β is obtained. If β is positive, then the dominant eigenvalue coincides with the largest eigenvalue and its corresponding eigenvector is taken. Otherwise, β is the least algebraic eigenvalue of $\mathbf{B}$ and a shift can be performed in $\mathbf{B}$. Power method is then performed over $\mathbf{B} + (|\beta|/2)\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. The method will converge to the desired eigenvector, since the eigenvectors of $\mathbf{B}$ are preserved, even with the shift. Thus, if the process described by (14) converges to an eigenvector corresponding to a negative dominant eigenvalue, the iterative process can be defined over $\mathbf{Ax} - (\mathbf{k}(\mathbf{k}^T\mathbf{x})/2m) + (|\beta|/2)\mathbf{I}$.

3.1.1. Computational Issues on the Implementation of the Fine-Tuning Stage. An important stage in Newman's spectral method implemented in this work is fine-tuning, which uses Kernighan-Lin method as postprocessing after each bisection performed, which is essential to assure the good quality of the communities obtained. In the work where fine-tuning is proposed [15], Newman does not define a methodology for its implementation, which is done in this section.

The fine-tuning stage must identify, among all nodes in the network, the one that, when moved between communities, causes the largest increase in the modularity. Direct use of (9) for each of the calculations leads to a high computational cost in the execution and, thus, Sun et al. propose an alternative solution. Instead of calculating the modularity of a division after a node is moved with (9), only the modularity variation δQ caused by moving such node is calculated, and the node associated with the highest value is then chosen. The variation δQ obtained by moving a node v_i can be described as

$$\delta Q = -\frac{\mathbf{s}_i}{m}\mathbf{B}_i^T\mathbf{s}, \tag{15}$$

Input: A network $G = (\mathbb{V}, \mathbb{E})$;
Two communities $\mathbb{C}''$ and $\mathbb{C}'''$;
A division vector $\mathbf{s}$ (7);
Number of nodes to be moved *eps*
Output: Two communities $\mathbb{C}''$ and $\mathbb{C}'''$;
A division vector $\mathbf{s}$ (after moving nodes)

(1) $\Delta Q \leftarrow 0$;
(2) $count \leftarrow 0$;
(3) $\mathbf{s}^* \leftarrow s$;
(4) $updated \leftarrow true$;
(5) **while** $count < eps$ **do**
(6) **if** *updated* **then**
(7) Calculate $\mathbf{B}^T\mathbf{s}$;
(8) $updated \leftarrow false$;
(9) **end**
(10) $\delta Q^* \leftarrow -1$;
(11) $\delta Q^*\_id \leftarrow 0$;
(12) **foreach** *element i in vector* $\mathbf{s}$ **do**
(13) **if** *element i still not moved* **then**
(14) Calculate δQ if i is moved from $\mathbb{C}''$ to $\mathbb{C}'''$ (or from $\mathbb{C}'''$ to $\mathbb{C}''$) with (15);
(15) **if** $\delta Q > \delta Q^*$ **then**
(16) $\delta Q^* \leftarrow \delta Q$;
(17) $\delta Q^*\_id \leftarrow i$;
(18) **end**
(19) **end**
(20) **end**
(21) Update vector $\mathbf{s}$ moving δQ^*_id from $\mathbb{C}''$ to $\mathbb{C}'''$ or from $\mathbb{C}'''$ to $\mathbb{C}''$;
(22) $count \leftarrow count + 1$;
(23) Mark element i as moved;
(24) **if** $\Delta Q + \delta Q > \Delta Q$ **then**
(25) $\mathbf{s}^* \leftarrow \mathbf{s}$;
(26) $updated \leftarrow true$;
(27) **end**
(28) **end**

Algorithm 4: Fine-tuning implemented in this work.

where $\mathbf{s}$ is calculated from the eigenvector obtained from a bisection in Newman's method.

Going beyond in the purpose of speeding up the execution of Newman's method, the implemented fine-tuning only performs the product $\mathbf{B}_i^T\mathbf{s}$ once at each node moving, in order to avoid computational waste. Moreover, the algorithm verifies if it is really necessary to update $\mathbf{B}_i^T\mathbf{s}$, which is only made when moving a node causes a positive variation in the modularity.

As defined for the other stages in Newman's spectral method, the value of matrix $\mathbf{B}$ is not explicitly calculated in the fine-tuning stage. Thus, the fine-tuning uses a strategy similar to the one used for the calculation of the dominant eigenvalue. Therefore, in any of the steps during the execution of Newman's spectral method, it is necessary to store the dense matrix $\mathbf{B}$, making the described tool very efficient regarding memory.

Algorithm 4 presents the steps for the execution of the fine-tuning stage considering the aspects previously mentioned.

Following the traditional approach for the fine-tuning stage, at each operation, the node whose moving causes the highest increase (or least decrease) in modularity is chosen, which is made for all the nodes of the network. That is, the value of *eps* is n (line 5 of Algorithm 4).

During the execution of the fine-tuning stage, there is an implicit ordination of each node regarding its attachment to the corresponding community. In the early iterations, fine-tuning will move the nodes which are less attached to their corresponding communities. After some permutations, only the nodes which are more attached to their communities will remain to be moved. It was verified by means of preliminary experiments that, at a certain point of the process, the probability is low that a node permutation results in a modularity gain.

This work proposes a reduction in the number of nodes to be moved in the fine-tuning stage. Two strategies were tested in this work: the permutation of 10% of the nodes and the permutation of 20% of the nodes in the fine-tuning stage. In order to do this, the values $n/10$ and $n/5$ were assigned to *eps* (line 5 of Algorithm 4). A comparative study of the results

obtained by the different parameters for the fine-tuning stage is presented in Section 4.

The computational tool for the implemented Newman spectral method with fine-tuning stage and the proposed variation is freely available for download in Github repository, in http://www.github.com/vfvieira/newman.

3.2. Computational Issues on CNM Implementation. The method of Clauset, Newman, and Moore (CNM) was implemented in ANSI C, without any external library. The variation of CNM proposed by Danon, Diaz, and Arenas (CNM-DDA) was also implemented.

In the work where CNM is proposed [12], the author uses a matrix **M** to store the modularity gain obtained when two generic communities $\mathbb{C}_a$ and $\mathbb{C}_b$ are combined. As the method proceeds successive communities joins, the lines of **M** related to $\mathbb{C}_a$ and $\mathbb{C}_b$ are successively combined. When the method is applied to real world networks, it is easy to notice that there is a great computational waste regarding memory usage (caused by the storage of null elements) and execution time (caused by the combination of null columns in both combined communities).

As presented in Section 2.3, Clauset et al. argue that when two communities are combined, it is only necessary to update the columns of **M** when at least one of the involved communities has nonzero elements and, to this end, the authors define (12) [1]. This solution avoids the network sparsity to increase the execution time when updating **M**.

In order to take advantage of the sparsity of **M** regarding memory usage as well, an immediate and naive idea is to use CSR storage scheme, as done in Newman's method. However, the CSR storage requires the nonzero structure to be preserved, which is not the case of **M**, where null elements may be filled when two communities are joined. A reasonable solution is to store each nonzero element of **M** in a balanced binary tree [1]. Particularly, this work uses AVL balanced binary trees to store the elements. Thus, insertions and deletions are performed in $O(\log n)$.

At each iteration of CNM method, a pair of communities $\mathbb{C}_a$ and $\mathbb{C}_b$ must be picked to be combined. A naive implementation must seek for the best pair of communities among each line of **M** (stored as an AVL tree), which can be computationally expensive. Thus, Clauset et al. [1] propose to use, for each line, a max-heap, which is able to return the largest element in $O(1)$ [38]. Further considerations regarding the computational complexity of CNM will be made in Section 3.3.

However, as highlighted by the authors [1], a simpler implementation using only one max-heap structure can be more efficient. In this case, when two generic lines, for instance, indexed by a and b (corresponding to $\mathbb{C}_a$ and $\mathbb{C}_b$, resp.) are joined, the method seeks for the communities $\mathbb{C}_c$ adjacent to $\mathbb{C}_a$ and $\mathbb{C}_b$ if its value in the heap is $\mathbf{M}_{ab}$ or $\mathbf{M}_{cb}$. If so, the value is updated. The present work implements the approach of using only one max-heap for the entire matrix **M**, which is performed with an array.

The computational tool for the implemented CNM method with DDA variation is freely available for download in Github repository, in http://www.github.com/vfvieira/cnm.

3.3. Comparison of Studied Methods. This section presents a comparative overview of the methods studied in this work, in order to provide a summary of references, main properties, positive and negative aspects, and computational complexity. This comparison aims to facilitate the discussion and the analysis of results presented in the next section.

Even though the methods studied in this work have the same purpose—modularity optimization—they are very distinct regarding the manner they use to do it, and some considerations can be made.

Newman's method works on a divisive approach based on spectral graph theory for community detection. It was presented by Newman and Girvan [6] and it is based on the optimization of a relaxed version of modularity function (presented by (3)) in which the solution vector **s** of (5) can assume any value (as in (7)). The solution of the optimization problem can be obtained by the eigenvector related to the largest eigenvalue of **B**, the modularity matrix (whose elements are defined by (4)).

The computational cost of Newman's spectral method can be calculated, primarily, as a combination of two elements: computation of eigenvalue **u** related to the largest algebraic eigenvalue of **B** (line 7 of Algorithm 1) and the depth of the execution, obtained by the number of divisions in the network. Finding an eigenvector of a dense matrix (for instance, **B**) costs $O(n^3)$ in a worst case scenario (n matrix-vector multiplications, each one running in $O(n^2)$). However, the calculation can be performed on a sparse matrix, which runs in $O(m + n)$, as presented by (14), reducing the cost of the operation to $O((m + n)n)$, if n iterations are needed to the eigenvalue to converge (which can be simplified to $O(mn)$). Considering that $m \sim n$ and $\log n$ divisions will be performed during the entire bisection process, the computational cost of Newman's method is, theoretically, $O(n^2 \log n)$.

Newman's spectral method is based on an elegant mathematical definition of the community detection problem and it is reported in the literature as a high quality method for community detection. However, when a very simple greedy approach is applied at each bisection, in a fine-tuning stage, the quality of the partitions found is significantly improved. Thus, the fine-tuning stage has also an important role in the algorithm for community detection implemented in this work. It is based on a variation of Kernighan-Lin method [34], originally designed for graph partitioning problems and adapted for community detection problem [6]. In order to do this, the constraints regarding the sizes of the groups are removed. The objective function is also adapted (from the minimization of the cut size to the maximization of the modularity).

In order to calculate the overall computational cost of the implemented method, the steps of the fine-tuning, presented in Algorithm 4, will be separately investigated. For each element i of **s** which has not been moved, δQ must be calculated by (15), which performs an inner product involving

$\mathbf{s}$ and the i^{-th} line of $\mathbf{B}$ (line 14 of Algorithm 4). However, in order to make a better use of the computational resources, this operation is replaced by the previous calculation of $\mathbf{B}^T\mathbf{s}$ (line 7 of Algorithm 4). Again, a matrix-vector multiplication involving $\mathbf{B}$ must be performed but, as defined by (14), it can be executed in a sparse structure and, thus, this operation is executed in $O(m)$. The product $\mathbf{B}^T\mathbf{s}$ must be executed only when ΔQ is updated, which may vary from once (when no swap increases the modularity value) to n times (when all swaps increase the modularity value). In a realistic scenario, a low number of swaps actually increase the quality of the partition, and the computational complexity of this stage can be stated as $O(m \log n)$. Also, calculation of δQ must be performed only for the elements which have not been moved yet, which decays at each iteration and can be considered as $\log n$. Considering, yet, that these operations must be performed *eps* times (where *eps* is the number of nodes swapped in the fine-tuning stage), the overall computational complexity of fine-tuning may vary from $O(eps((m) + (\log n)))$ to $O(eps((mn) + (\log n)))$ (or $O(eps((m \log n) + (\log n)))$ in a quite realistic scenario). The computational time for Newman's method with the fine-tuning stage is $O([n^2 \log n] + O[eps((m \log n) + (\log n))])$. Fine-tuning does not increase the complexity order of the method but, still, it increases the execution time, setting a trade-off between complexity and quality.

An agglomerative greedy heuristic method for community detection was proposed by Newman [12], which starts with n unary communities and, at each step, joins the pair of communities that causes the largest increase in the modularity Q. Using a matrix to store the values, each of the n steps of the algorithm is executed in $O(m + n)$, resulting in a computational complexity of $O(n + m)$. Based on Newman's heuristic method, Clauset, Newman, and Moore (CNM) propose a method [1] which, making use of proper data structures to store the values, can substantially reduce the cost of the execution. CNM method is not as elegant as Newman's method, but it shows lower execution complexity when compared to Newman's method, which enables it, theoretically, to be applied to larger networks. As previously reported, this work uses AVL binary trees to store the values, in which an average search can be executed in $\log n$. Joining two communities (line 6 of Algorithm 3) can, then, be performed in $O(\log^2 n)$. However, one of the authors argues that this complexity is underestimated in practical scenarios and the implementation of CNM method behaves as $O(n \log^2 n)$ only if the agglomerations are performed in a balanced way (http://cs.unm.edu/~aaron/blog/archives/2007/02/fastmodularity.htm).

The methodology proposed by Danon, Diaz-Guilera, and Arenas (CNM-DDA) [24] modifies the CNM method in order to prioritize the combination of balanced communities. As reported by the authors, this modification harms the modularity values in the first iterations, since the greedy characteristic of the CNM is affected in favour of balanced communities. The modification of DDA requires the calculation of the modularity gain, which, at a first glance, appears to increase the execution time. However, as it will be noticed in Section 4, the balanced combinations reduce significantly the execution time.

4. Experiments and Discussion

This section presents the results obtained by the application of the methods described in Section 3 to a set of 24 real world networks, frequently used as benchmarks in network community structure studies. All the experiments were executed in a PC laptop with an Intel Core i7 2.3 GHz 64 bit CPU and 8 Gb RAM running Ubuntu 12.10. Table 1 shows a descriptive summary of the main features of the networks: number of nodes (n), number of edges (m), average degree ($\widehat{d}$), and average clustering coefficient ($\widehat{cc}$). A brief description of each of the networks is also presented.

The experiments performed refer to the execution of community detection methods, following the methodology presented in Section 3. The method of Clauset, Newman, and Moore (CNM) was explored due to its quality, efficiency, and high number of citations in the literature considering heuristic methods for community detection in large scale networks. Moreover, the modification of Danon, Diaz, and Arenas (CNM-DDA) was also implemented, due to the quality of the results reported in its original work [24]. Newman's spectral method (Newman), reported in the literature as a method which results in high quality partitions, was also implemented in combination with the fine-tuning stage (based on Kernighan-Lin method), using different values of nodes swaps (Newman-FT, Newman-FT10%, and Newman-FT20%).

Table 2 presents the quality of partitions, assessed by modularity Q, obtained by the different algorithms implemented and applied to the tested networks. The best result found for each network is highlighted in bold font. The number of communities found by each method is also presented in Table 2.

The analysis of Table 2 allows some remarks to be made. At first glance, it is possible to notice that the worst results are obtained for CNM and Newman's methods without modifications. When the modifications are applied (DDA and fine-tuning, resp.), on the other hand, the results exhibit excellent modularity values. Most of the best results are obtained by Newman-FT and, also, some of them are found by CNM-DDA. These results are coherent to those found in the literature, as presented in the works of Newman [15] and Danon et al. [24], where authors strongly suggest using the modifications.

When the results obtained by the spectral method and its variations are compared, the strategy of reducing the number of nodes moved in the fine-tuning stage (proposed in this work) shows itself very promising. In some cases, moving only 10% of the nodes in the fine-tuning stage (Newman-FT10%) is enough to provide results as good as those provided when 100% of the nodes are moved (Newman-FT). When the number of moved nodes is increased to 20% of the nodes (Newman-FT20%), the obtained modularity, in almost all the tested networks, is the same to the value obtained when 100% of the nodes are moved. In other words, in most cases, the

Table 1: Main features of the networks: number of nodes n, number of edges m, average degree $\hat{d}$, and average clustering coefficient $\widehat{cc}$.

Network	n	m	$\hat{d}$	$\widehat{cc}$	Type
Karate[1]	34	78	4.59	0.57	Social network
Dolphins[1]	62	159	5.13	0.26	Dolphins network
Jazz[1]	198	2742	27.70	0.62	Social network
Football[1]	115	613	10.66	0.40	Games network
Adjnoun[1]	112	425	7.59	0.17	Words network
Les Mis.[1]	77	254	6.60	0.58	Coappearance network
Polbooks[1]	105	441	7.59	0.49	Copurchasing network
Email[2]	1133	5451	9.62	0.22	Communication network
C.Elegans[2]	453	2040	9.01	0.65	Metabolic network
Netscience[1]	1461	2742	3.75	0.70	Coauthorship network
Keys[2]	10680	24316	4.55	0.27	Information network
Cond-Mat[1]	16264	47594	5.85	0.64	Coauthorship network
Cond-Mat03[1]	30460	120029	7.88	0.65	Coauthorship network
Amazon0302[3]	262111	899792	6.87	0.42	Copurchasing network
BerkStan[3]	685230	6649470	19.41	0.60	Web documents network
CA-AstroPh[3]	18772	198110	21.11	0.63	Coauthorship network
CA-CondMat[3]	23133	93497	8.08	0.63	Coauthorship network
CA-GrQc[3]	5242	14496	5.53	0.53	Coauthorship network
CA-HepPh[3]	12008	118521	19.74	0.61	Coauthorship network
CA-HepTh[3]	9877	25998	5.26	0.47	Coauthorship network
Cit-HepPh[3]	34546	420921	24.37	0.28	Citation network
Cit-HepTh[3]	27770	352324	25.37	0.31	Citation network
Com-Amazon[3]	334836	925872	5.53	0.40	Copurchasing network
Com-Youtube[3]	1134890	2987624	5.26	0.08	Social network

[1]Downloaded from http://www-personal.umich.edu/~mejn/netdata/.
[2]Downloaded from http://deim.urv.cat/~aarenas/data/welcome.htm.
[3]Downloaded from http://snap.stanford.edu/data/.

permutation of more than 80% of the nodes in the fine-tuning stage only causes a huge waste of time and computational resources.

From the analysis of Table 2, it can be noticed that the results obtained by Newman's method with fine-tuning (even when only 10 or 20% of the nodes are moved) are always quite close to the best results found. Thus, it is a reasonable choice to use Newman's spectral method (Newman-FT) or one of its variations (Newman-FT10% or Newman-FT20%).

It is interesting to note that there is a tendency of good results obtained by the heuristic methods (CNM and CNM-DDA) for networks in which the modularity values are higher, that is, networks with a very well defined community structure.

This behaviour can be explained by the degeneracy characteristics shown by the modularity function, as presented by Good et al. [18]. In other words, there are a typically exponential number of distinct partitions with modularity values close to the optimum and this number tends to increase when the communities are better defined.

Newman's spectral method shows an interesting behaviour when applied to the largest networks (BerkStan and Com-Youtube). The quality of the partitions obtained when fine-tuning is applied is much higher if compared to the method without fine-tuning. Such results can be explained when the number of communities found is also considered. For instance, considering BerkStan, $Q = 0.3242$ and 5 communities are found by Newman's method without fine-tuning. Moreover, $Q = 0.9189$ and 105 communities are found by Newman-FT. The comparison of these results suggests that the execution of Newman's method without fine-tuning is prematurely interrupted, resulting in a partition with only 5 communities. A similar analysis can be performed for Com-Youtube network.

It is also worth noticing from Table 2 that the number of communities found by the different versions of the spectral method is not related to the quality of the partitions. In some cases, more communities are found when the modularity is higher (e.g., Football, Email, and Cit-HepTh). For other networks, less communities are found when the modularity increases (e.g., CA-AstroPh and CA-CondMat). In most cases, however, both behaviours happen for the same network; that is, for most networks, the quality of partitions is not related to the number of communities found by each variation of the spectral method (e.g., for Amazon0302, $Q = 0.6931$ and 450 communities are found by Newman's method without fine-tuning; $Q = 0.8486$ and 325 communities are found by Newman-FT; $Q = 0.8369$ and 332 communities are found by Newman-FT10%). Still, a correlation between modularity and number of communities found can not be observed when CNM and CNM-DDA are compared.

Table 2: Modularity Q obtained for each network/number of communities found.

Network	Newman	Newman-FT	Newman-FT10%	Newman-FT20%	CNM	CNM-DDA
Karate	0.3934 /4	**0.4188** /4	0.4097 /4	**0.4188** /4	0.3806 /3	**0.4188** /4
Dolphins	0.5090 /5	**0.5143** /6	**0.5143** /6	**0.5143** /6	0.4954 /4	0.5067 /4
Jazz	0.3936 /3	**0.4422** /4	**0.4422** /4	**0.4422** /4	0.4389 /4	0.4355 /3
Football	0.4883 /8	**0.6009** /10	**0.6009** /10	**0.6009** /10	0.5704 /6	0.5712 /8
Adjnoun	0.2450 /9	0.2915 /7	0.2900 /7	0.2915 /7	**0.2929** /7	0.2836 /6
Les Mis.	0.5314 /6	**0.5443** /6	**0.5443** /6	**0.5443** /6	0.5005 /5	0.5327 /5
Polbooks	0.4650 /4	**0.5246** /4	**0.5246** /4	**0.5246** /4	0.5019 /4	0.5098 /5
Email	0.4870 /7	**0.5526** /10	**0.5526** /10	**0.5526** /10	0.5065 /15	0.5211 /9
C.Elegans	0.3430 /7	**0.4233** /7	0.4168 /9	**0.4233** /7	0.4094 /10	0.4202 /12
Netscience	0.9188 /53	0.9442 /87	0.9378 /69	0.9419 /79	0.9555 /276	**0.9585** /276
Keys	0.7679 /94	0.8505 /77	0.8447 /80	0.8505 /77	0.8515 /193	**0.8570** /153
Cond-Mat	0.7397 /123	**0.7935** /147	0.7918 /139	0.7930 /139	0.7860 /895	0.7902 /796
Cond-Mat03	0.6299 /143	**0.7187** /117	0.7163 /122	**0.7187** /117	0.6675 /1268	0.6906 /977
Amazon0302	0.6931 /450	**0.8486** /325	0.8369 /332	0.8458 /348	0.8215 /1652	0.8402 /494
BerkStan	0.3242 /5	**0.9189** /105	0.9048 /100	0.9060 /106	—	—
CA-AstroPh	0.4876 /61	**0.5921** /34	0.5868 /53	**0.5921** /34	0.4944 /443	0.5510 /333
CA-CondMat	0.5955 /133	**0.6882** /100	0.6833 /102	**0.6882** /100	0.6320 /877	0.6757 /627
CA-GrQc	0.7814 /79	**0.8337** /80	0.8256 /75	**0.8337** /79	0.8043 /419	0.8316 /396
CA-HepPh	0.5344 /7	**0.6403** /51	0.6383 /52	**0.6403** /51	0.5798 /432	0.6011 /323
CA-HepTh	0.6493 /106	0.7310 /73	0.7091 /110	**0.7352** /89	0.7138 /551	0.7297 /475
Cit-HepPh	0.5658 /24	**0.7125** /25	0.6850 /55	0.7074 /34	0.5261 /201	0.5735 /85
Cit-HepTh	0.4868 /11	**0.6244** /34	0.4868 /11	**0.6244** /34	0.5058 /285	0.5593 /177
Com-Amazon	0.8682 /488	0.8739 /390	0.8694 /463	0.8694 /496	0.8682 /1476	**0.8781** /688
Com-Youtube	0.3616 /11	**0.6930** /333	0.6812 /316	**0.6930** /333	—	—

Table 2 also shows that CNM and CNM-DDA tend to provide partitions with a larger number of communities when compared to Newman's spectral method and the difference is more significant in larger networks. Such behaviour can be explained by the properties of the methods. As Newman's method is performed in a top-down manner, the first partition found by the method has only one community (the entire network) and the number of communities increases as the method is executed. Thus, the partitions found by the method during its execution tend to show a low number of

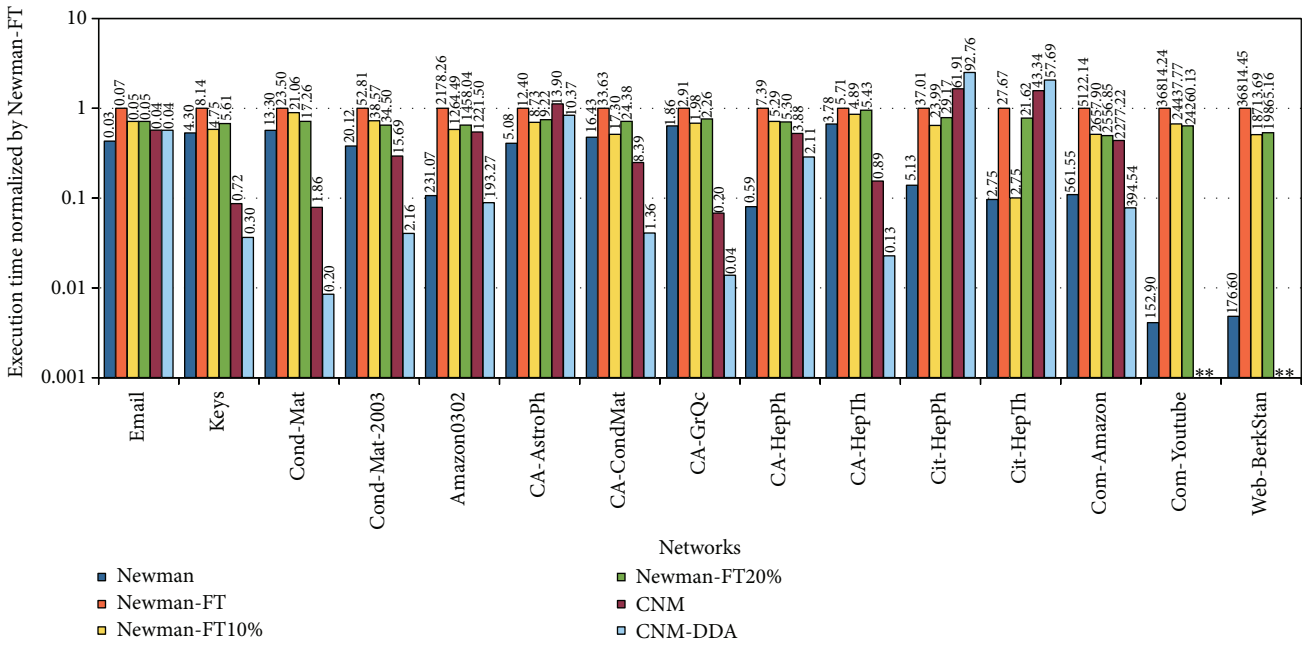

FIGURE 1: Execution time for each network with each method normalized by the time observed for Newman-FT in log scale. Actual execution time (in seconds) is also presented above each bar. CNM and CNM-DDA could not be executed for Com-Youtube and Web-BerkStan networks.

communities. The opposite occurs with CNM and CNM-DDA. As it performs in a bottom-up manner, the partitions found during the execution show a large number of communities.

A broader investigation of the methods is performed considering the execution time for each of the 24 tested networks. For the networks with up to 1000 nodes (Karate, Dolphins, Jazz, Football, Adjnoun, Les Miserables, and Polbooks), the executions were performed almost instantly, in approximately 10^{-2} seconds. A comparison of the implemented methods for community detection considering the execution time of these networks would be quite poor and inconclusive. Thus, this discussion is basically focused on the larger networks.

Figure 1 allows the analysis of the execution time of the implemented methods for the networks presented in Table 1 with more than 1000 nodes. In order to provide a comparative analysis of the methods, the results shown in Figure 1 (y-axis) were normalized in respect to the execution time observed for Newman's spectral method with traditional fine-tuning (Newman-FT), which takes more time to be executed in most of the executions performed. The results are presented in log scale. For reference purpose, actual execution time of each method for each network (in seconds) using the computational environment described at the beginning of this section is also exhibited.

At first glance, it can be noticed that CNM-DDA shows a good performance in a wide number of networks. In particular, the method is executed quite fast in networks up to ~30000 nodes. Thus, when this result is combined with other results in the literature, which suggest that CNM-DDA leads to good modularity values, the method can be considered as a reasonable choice for processing medium scale networks and a fast response is needed.

It can be also noticed from Figure 1 that DDA modification considerably improves the CNM method regarding the execution time (the same consideration can be made regarding modularity values, when Table 2 is observed) and these results are coherent with those reported in the literature and discussed in Section 3.3. The unbalanced union of communities in CNM method considerably harms the execution time, making it reasonable to apply the DDA variation to real world networks.

Still regarding time complexity, it is worth saying that the theoretical computational complexity of Newman's method, as reported in the literature, is $O(n^2 \log n)$. The term $\log n$ is used to represent the expected height for the division tree, which is much lower than the number of nodes n and makes Newman's method to behave, in practice, very similarly to CNM method.

Another interesting point to notice is that the execution time for Newman's method, apart from depending on the network size (given by n and m) and the height of the divisions tree, also depends on the number of operations performed at each division. In other words, the number of divisions and the sizes of the subnetworks to be divided at each step of the execution directly affect the total time. As a result, it may occur that, for the same network, Newman's method with 10% of the nodes swapped in the fine-tuning stage shows a higher execution time when compared to Newman's method with 20% of the nodes swapped. This can be observed in Figure 1 for the networks Cond-Mat, Cond-Mat03, Com-Amazon, and Com-Youtube.

Indeed, Newman's method (without the fine-tuning stage) shows good results regarding the execution time but, on the other hand, it leads to low modularity values when compared to other methods. Fine-tuning stage, as shown in Table 2, significantly increases the modularity value and, on the other hand, increases the execution time. Thus, Newman's method can be considered as an elegant way of finding communities, which can be, then, refined with Kernighan-Lin method.

Newman's and CNM-DDA methods show low execution time for a wide range of the studied networks. However, a more careful analysis reveals that, for some networks (BerkStan, Com-Youtube, Amazon0302, and Cit-HepTh), the comparison of the execution time and the quality of the partitions presented in Table 2 suggest that the execution of Newman's method is, indeed, prematurely finished, as suggested by the number of communities found and previously discussed. The reduction in the execution time, then, brings as consequence a great harm in modularity.

From the combined analysis of Table 2 and Figure 1, a clearer comparison of the studied methods can be made. It can be noticed that, even with a higher execution time, when the implementation of Newman's method is combined with the fine-tuning stage, based on Kernighan-Lin method, the high modularity values obtained pose an interesting trade-off between time × modularity, specially when the strategies of moving only 10% (Newman-FT10%) and 20% (Newman-FT20%), proposed in this work, are considered. It can be also argued that, even in cases where Newman-FT10% and Newman-FT20% present higher execution times, its ratio in relation to the best time obtained does not vary in scale (except in the previously mentioned cases where Newman's method without fine-tuning finishes prematurely) and the strategy of reducing the number of nodes swapped in the fine-tuning stage can be considered quite reasonable.

5. Conclusions

This work presented a study of some of the most discussed methods for community detection based on modularity: Newman's spectral method with a fine-tuning stage, the method of Clauset, Newman, and Moore (CNM), and the variation of Danon, Diaz, and Arenas (DDA) for the CNM method. The computational issues of the implemented methods were deeply analysed, in order to provide a set of high performance tools for community detection regarding execution time and memory usage. The methods were applied to a set of large scale real networks. The implementation of the fine-tuning stage proposed in this work considers a reduction in the number of swapped nodes, which enables an up to 50% faster execution without harming the quality of the partitions obtained.

The application of the implemented methods to a set of 24 networks with different features shows that the modularity obtained is consistent with the results found in the literature. Yet, for small networks (up to ~10000 nodes), the execution is quite fast, which suggest that the application of the methods is quite appropriate in contexts where an immediate response is needed.

Newman's method with traditional fine-tuning shows the best modularity value for almost all tested networks. However, even in situations where other methods show higher modularity values, the results obtained by Newman's method with fine-tuning are close to the best result. It can be also noticed that CNM and CNM with the variation of DDA show better modularity values, compared to Newman's method when the community structure is better defined, that is, when the modularity value is high.

When Newman's method with traditional fine-tuning is compared to the proposed variation, considering only 10% and 20% of nodes swap, it can be noticed that, even reducing the number of nodes moved between communities, the quality of the communities found remains very similar. On the other hand, the variation allows a faster execution and its use is very reasonable.

An analysis on the computational complexity of Newman's method and CNM method allows the observation that even though they are theoretically distinct, the execution time is very similar. This happens due to some reasons. The first one is the great sensibility of CNM to the unbalancing of the union tree. Another reason is that, on both methods, the height of the execution tree impacts the execution time and it is much lower in Newman than it is in CNM.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors are grateful to the agencies CNPq, CAPES, and FAPERJ for their financial support.

References

[1] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, Article ID 066111, 2004.

[2] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3–5, pp. 75–174, 2010.

[3] R. Guimerà and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, no. 7028, pp. 895–900, 2005.

[4] Y. I. Leon-Suematsu and K. Yuta, "Framework for fast identification of community structures in large-scale social networks," in *Data Mining for Social Network Data*, vol. 12 of *Annals of Information Systems*, pp. 149–175, Springer, New York, NY, USA, 2010.

[5] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proceedings of the 19th International World Wide Web Conference (WWW '10)*, pp. 631–640, ACM, New York, NY, USA, April 2010.

[6] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E: Statistical,*

Nonlinear, and Soft Matter Physics, vol. 69, no. 2, Article ID 026113, 2004.

[7] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," in *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics (MDS '12)*, vol. 3, pp. 1–8, ACM, Beijing, China, 2012.

[8] F. Moradi, T. Olovsson, and P. Tsigas, "An evaluation of community detection algorithms on large- scale email traffic," in *Experimental Algorithms*, R. Klasing, Ed., vol. 7276 of *Lecture Notes in Computer Science*, pp. 283–294, Springer, Berlin, Germany, 2012.

[9] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, Article ID 026113, 2004.

[10] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002.

[11] L. C. Freeman, "Centrality in social networks conceptual clarification," *Social Networks*, vol. 1, no. 3, pp. 215–239, 1978.

[12] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 69, no. 6, Article ID 066133, 2004.

[13] G. Agarwal and D. Kempe, "Modularity-maximizing graph communities via mathematical programming," *European Physical Journal B*, vol. 66, no. 3, pp. 409–418, 2008.

[14] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 72, no. 2, Article ID 027104, 2005.

[15] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 23, pp. 8577–8582, 2006.

[16] F. R. K. Chung, *Spectral Graph Theory*, American Mathematical Society, 1996.

[17] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 1, pp. 36–41, 2007.

[18] B. H. Good, Y.-A. de Montjoye, and A. Clauset, "Performance of modularity maximization in practical contexts," *Physical Review E*, vol. 81, no. 4, Article ID 046106, 2010.

[19] A. Lancichinetti and S. Fortunato, "Limits of modularity maximization in community detection," *Physical Review E*, vol. 84, no. 6, Article ID 066122, 2011.

[20] P. Ronhovde and Z. Nussinov, "Multiresolution community detection for megascale networks by information-based replica correlations," *Physical Review E*, vol. 80, no. 1, Article ID 016109, 2009.

[21] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral, "Modularity from fluctuations in random graphs and complex networks," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 70, no. 2, Article ID 025101, 2004.

[22] A. Kehagias, "Bad communities with high modularity," Computing Research Repository abs/1209.2678, 2012.

[23] S. Cafieri, P. Hansen, and L. Liberti, "Improving heuristics for network modularity maximization using an exact algorithm," *Discrete Applied Mathematics*, vol. 163, part 1, pp. 65–72, 2014.

[24] L. Danon, A. Diaz-Guilera, and A. Arenas, "Effect of size heterogeneity on community identification in complex networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2006, Article ID P11010, 2006.

[25] H. Li, H. Jiang, R. Barrio, X. Liao, L. Cheng, and F. Su, "Incremental manifold learning by spectral embedding methods," *Pattern Recognition Letters*, vol. 32, no. 10, pp. 1447–1455, 2011.

[26] M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, Article ID 036104, 2006.

[27] V. da Fonseca Vieira and A. G. Evsukoff, "A comparison of methods for community detection in large scale networks," *Studies in Computational Intelligence*, vol. 424, pp. 75–86, 2013.

[28] K. Wakita and T. Tsurumi, "Finding community structure in mega-scale social networks," *Analysis*, vol. 105, no. 2, article 9, 2007.

[29] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, Article ID P10008, 2008.

[30] M. E. J. Newman, *Networks: An Introduction*, Oxford University Press, Oxford, UK, 1st edition, 2010.

[31] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, vol. 11, Article ID 033015, 2009.

[32] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, Article ID 066111, 2004.

[33] M. E. J. Newman, "Coauthorship networks and patterns of scientific collaboration," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 1, pp. 5200–5205, 2004.

[34] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *The Bell System Technical Journal*, vol. 49, no. 1, pp. 291–307, 1970.

[35] S. Balay, J. Brown, K. Buschelman et al., "PETSc users manual," Tech. Rep. ANL-95/11-Revision 3.3, Argonne National Laboratory, 2012.

[36] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, 2nd edition, 2003.

[37] M. T. Heath, *Scientific Computing: An Introductory Survey*, McGraw-Hill, New York, NY, USA, 2nd edition, 2002.

[38] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, vol. 2, The MIT Press, 2001.