Scalable Spectral Clustering for Overlapping Community Detection in Large-Scale Networks

Hadrien Van Lierde, *Graduate Student Member, IEEE,* Tommy W. S. Chow, *Fellow, IEEE,* and Guanrong Chen, *Fellow, IEEE*

Abstract—While the majority of methods for community detection produce disjoint communities of nodes, most real-world networks naturally involve overlapping communities. In this paper, a scalable method for the detection of overlapping communities in large networks is proposed. The method is based on an extension of the notion of normalized cut to cope with overlapping communities. A spectral clustering algorithm is formulated to solve the related cut minimization problem. When available, the algorithm may take into account prior information about the likelihood for each node to belong to several communities. This information can either be extracted from the available metadata or from node centrality measures. We also introduce a hierarchical version of the algorithm to automatically detect the number of communities. In addition, a new benchmark model extending the stochastic blockmodel for graphs with overlapping communities is formulated. Our experiments show that the proposed spectral method outperforms the state-of-the-art algorithms in terms of computational complexity and accuracy on our benchmark graph model and on five real-world networks, including a lexical network and large-scale social networks. The scalability of the proposed algorithm is also demonstrated on large synthetic graphs with millions of nodes and edges.

Index Terms—Community detection algorithm, overlapping communities, real-world network, spectral clustering, Normalized Cut.

1 Introduction

The past two decades have witnessed a growing interest in the analysis of complex networks with various applications. Such networks are modelled as graphs in which nodes represent network agents and edges represent interactions among agents. A common property of large networks, including social networks, is that agents tend to form groups with large numbers of connections within each group but small numbers of connections between groups. Such groups are referred to as communities or clusters of nodes.

Most of the research in community detection focuses on disjoint clusters [1]. Popular algorithms include hierarchical clustering [2], spectral clustering [3], methods based on edge centrality measures [4] and modularity-based methods [5]. However, in various real-world applications, the assumption that any agent of a network only belongs to a single group is not valid. For instance, an individual in a social network may have different hobbies and areas of interest, and hence the individual may be a member of different social groups. Thus, some methods were proposed for detecting overlapping communities, allowing multiple community memberships for agents [6].

The majority of existing methods of overlapping community detection focus on finding overlapping communities of nodes. This includes algorithms such as the clique percolation method [7], methods based on local optimization of a community fitness measure [8], [9], and methods based on fuzzy clustering [10], [11], [12]. In contrast, edge-based community detection algorithms are proposed in [13], [14], [15]. They first find disjoint communities of edges, then a node is regarded as belonging to a community l if one of its incident edges belongs to the edge community l. Thus, the resulting node communities may have nodes in common. The advantage of this approach is its flexibility since any algorithm detecting disjoint communities can be applied for edge partitioning. However, there are several major weaknesses to these edge-based methods. First, none of these algorithms provide ways to incorporate prior knowledge about the number of communities that each node is likely to belong to, even when it is available. Second, they do not rely on a clear definition of node community, as pointed out in [1]. They apply various community detection algorithms to edge-based graphs without providing clear theoretical interpretation of the resulting overlapping communities of nodes. Third, the optimization of the associated objective functions (e.g., the modularity) relies on greedy approximations, or on costly methods such as Simulated Annealing. These optimization methods are prone to getting stuck in local optima. Finally, these methods have a high computational cost when the graph becomes denser. Indeed, these algorithms rely on the explicit representation of an underlying graph over edges (e.g., the line graph), whose adjacency matrix has more nonzero elements than that of the original graph. For instance, for a k-regular graph, the number of nonzero elements of the adjacency matrix is $O(k|V|)$ for the original graph but $O(k^2|V|)$ for the edge-based line graph. Hence, the cost of applying standard community detection algorithms (e.g., based on modularity maximization) is higher on an edge-based graph than on the original graph.

To alleviate the above problems, this paper proposes a

• *The authors are with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, China (e-mail: hvanlierd2-c@my.cityu.edu.hk, eetchow@cityu.edu.hk, eegchen@cityu.edu.hk).*

Manuscript received November 30, 2017

new edge-based method that extends the effective spectral clustering algorithm. Spectral clustering provides approximate solutions to the Normalized Cut (Ncut) minimization problem in order to extract non-overlapping clusters of nodes [3]. The spectral clustering algorithm first computes extremal eigenvalues of the graph Laplacian matrix. Then, it extracts communities of nodes by clustering the entries of the corresponding eigenvectors using the k-means algorithm. The spectral clustering method is popular for its computational efficiency and its theoretical connection with Ncut. Some methods were proposed to extend spectral clustering to the detection of overlapping clusters. However, these works essentially replace the k-means step with fuzzy clustering algorithms, such as the c-means algorithm [16] or the Gaussian mixture model [17]. But they do not extend the definition of Ncut to overlapping communities. To the best of our knowledge, the only approach that uses Ncut for finding overlapping communities was perhaps [18]. However, it merely extends the Laplacian-based definition of Ncut by appending constraints on cluster indicator variables.

The method proposed in this paper relies on an edge-based extension of the theoretical notion of Ncut. New definitions of cuts and volumes of communities in the overlapping case are first introduced, providing a stronger theoretical foundation in comparison with the previous extensions of spectral clustering or Ncut [16], [17], [18]. A scalable spectral clustering algorithm is then proposed to solve the associated cut minimization problem, thereby extracting the overlapping communities. In addition, a hierarchical version of our algorithm is introduced to detect the number of clusters automatically. Finally, we introduce a way to take into account prior knowledge about the number of communities that an agent is likely to belong to. This prior knowledge can be obtained from metadata[1], or it can be based on topological characteristics of the graph. When such information is available, it is incorporated as node weights in our proposed spectral algorithm, which enhances the quality of the resulting communities.

The proposed approach alleviates the above-mentioned weaknesses of existing edge-based clustering algorithms [13], [14], [15]. Indeed, it relies on a new theoretical definition of overlapping communities based on Ncut. Moreover, by introducing node weights, we provide a way to include prior information about the number of communities each node is likely to belong to. The space and time complexities of our method are also proved to be small compared to existing edge-based methods. Indeed, it relies on the computation of the second smallest eigenvalue of a graph-related matrix with an explicit factorization which is used to accelerate the computation of the eigenvalue. This matrix contains $O(|E|)$ nonzero elements, where $|E|$ is the number of edges in the original graph. Hence, it has only half the number of nonzero elements of the adjacency matrix of the original graph. As a result, the time and space complexities of our algorithm are identical to that of the efficient node-based spectral clustering algorithms.

In order to test the efficiency of our algorithm, we propose a new benchmark graph model based on an extension of the stochastic blockmodel [19]. The performance of our algorithm on this benchmark graph is analyzed, showing that it outperforms six state-of-the-art methods for the detection of overlapping clusters, including the clique percolation method [7], two edge-based clustering algorithms [13], [14], a fuzzy clustering scheme [16], an approach based on local optimization [9], and an approach extending the k-means algorithm for overlapping community detection [18]. Also, the scalability of our algorithm is demonstrated, showing that it is able to process a graph of one million nodes and ten million edges in about a minute with a `Matlab` program running on a standard personal computer. Finally, our algorithm is tested on five real-world networks with ground-truth communities, namely Zachary's karate club, the Word Association network, Amazon's co-purchasing network, and two large-scale social networks derived from DBLP and YouTube platforms, respectively. Our approach is shown to outperform all other comparable algorithms on these networks.

The main contributions of the present paper are four-fold: (1) the introduction of the *Vertex Reinforced Overlapping Normalized Cut* (VRONcut) problem for the detection of overlapping communities in an undirected graph with weighted edges and nodes, (2) the formulation of a spectral clustering algorithm for finding an approximate solution to the VRONcut problem and for automatically detecting the number of communities, (3) the introduction of a new benchmark graph model based on an extension of the stochastic blockmodel, which enables the generation of networks with overlapping communities in various forms, and (4) a thorough comparative analysis with the state-of-the-art methods of overlapping community detection, which demonstrates the efficiency and the scalability of the proposed approach.

The paper is structured as follows. In Section 2, related methods of overlapping community detection are presented. In Section 3, the new definition of VRONcut is presented and the associated spectral clustering algorithm is derived. In Section 4, the performance of our algorithm on a benchmark graph model is analyzed and compared with other methods. In Section 5, we present the results of our algorithm when it is applied to five real-world networks. The last section concludes the investigation.

2 Related work

According to [6], existing approaches for the detection of overlapping communities generally fall into four categories, which include methods based on edge partitioning, clique-based methods, methods based on local improvement of a quality function, and fuzzy clustering algorithms. Clique-based methods, such as the Clique Percolation Method [7], [20], [21], treat the communities as overlapping complete subgraphs or cliques. While these methods are efficient when the communities are densely connected, they rely on a pattern matching principle instead of a more flexible definition of community [6]. Hence, they often fail when the connections are sparse. Moreover, their computational cost is high due to the lengthy process of finding cliques [6]. Some other approaches are based on local improvement

1. For instance, the number of groups or number of areas of interest of an agent in a social network is considered as an indication of the likelihood for this agent to belong to several communities.

of a quality criterion [8], [9]. An initial set of communities is computed, and then local modifications of clusters are performed in order to improve a quality measure such as the balance between inner and outer connections of the communities [8], [9]. A critical issue is the necessity of finding a valid initial set of seed nodes, which is often computationally expensive and has a significant impact on the quality of the resulting communities [8], [22]. Methods based on fuzzy clustering produce a probability distribution over clusters for each node. Fuzzy memberships are either inferred based on a generative model [10], or they are obtained by solving an optimization problem expressing the quality of the communities. The latter case involves algorithms such as simulated annealing [11], nonnegative factorization of the adjacency matrix [12], or computation of low-rank decompositions of the adjacency matrix [23], [24]. These algorithms have a high computational cost, and they require prior knowledge on the number of communities. This requirement is impractical for many application problems.

Recent approaches suggest detecting disjoint groups of edges rather than nodes and then finding overlapping communities of nodes based on the partitioning of the edges [13]. First, a graph depicting similarities between edges is defined (e.g., the line graph [14], [15], or a graph based on the Jaccard similarity [13]). Then, disjoint edge communities are extracted by applying standard community detection algorithms to this edge-based graph (e.g., the agglomerative clustering algorithm [13], or a modularity-based method [14]). Finally, a node belongs to a node community l if at least one of its incident edges belongs to the edge community l. An advantage of this edge-based approach is that many simple node clustering algorithms are available to detect edge clusters. However, as mentioned in Section 1, the main drawbacks of these approaches are their heavy time and space complexities, their inability to incorporate prior information about the number of communities per node, and the lack of a clear definition of the node communities they aim at.

Finally, there are still some methods that do not fall into the above categories such as the game theoretic approach proposed in [25], which represents nodes as agents and each community as a Nash equilibrium, a speaker-listener model [26], in which community labels are spread according to pairwise relationships between speakers and listeners, or a kernelized version of the overlapping k-means algorithm [18]. The last method extends the k-means algorithm by allowing samples to be assigned to multiple clusters. A weighted kernel extension of this method is then proposed to tackle the problem of overlapping community detection in graphs. The method can be viewed as an extension of the Laplacian-based definition of Ncut for disjoint community detection.

3 Spectral Clustering Algorithm for Extracting Overlapping Communities

In this section, we introduce our main algorithm for the detection of overlapping communities. We use the terms *community* and *cluster* interchangeably. All the graphs analyzed in this paper are connected, undirected and possibly weighted. We refer to such a graph as a triplet $G = (V, E, w)$, where V is a set of nodes, $E \subseteq V \times V$ is a set of edges and $w \in \mathbb{R}_+^{|E|}$ is a vector of positive edge weights. We define the incidence matrix $M \in \{0,1\}^{|V|\times|E|}$ such that $M_{ie} = 1$ if edge e is incident to node i. When the graph is unweighted, we use the simplified notation $G = (V, E)$. When additional node weights are included, we refer to such a graph as a quadruplet $G = (V, E, w, \phi)$, where $\phi \in \mathbb{R}_+^{|V|}$ is a vector of positive node weights. We also denote the weight matrix by $W \in \mathbb{R}_+^{|V|\times|V|}$ such that W_{ij} is the weight of edge (i, j).

3.1 Motivation and method outline

Traditional partitioning algorithms based on Ncut minimization [27] partition nodes of a graph into disjoint clusters by minimizing the number of "inter-cluster edges". However, this approach fails in cases such as the one illustrated by Figure 1, in which some nodes, referred to as *bridge nodes*, lie in the overlap between communities. A method proposed in [13] addresses this issue by first extracting disjoint communities of edges, and then deriving overlapping clusters of nodes from edge communities. Based on this idea, we propose a new edge-based version of Ncut. We extract disjoint clusters of edges by minimizing the proportion of nodes lying in the overlap between communities, i.e. nodes with edges belonging to different clusters, as shown in Figure 1. We first focus on the problem of extracting two communities from a graph. Then, we extend the algorithm to the detection of any number of communities.

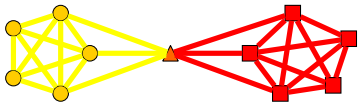


Figure 1. Graph consisting of two 5-cliques both connected to a central node. Colours of edges denote their natural partitioning into two non-overlapping clusters. Node colours and shapes denote the overlapping communities: Community 1 (circles and triangle), Community 2 (squares and triangle) and overlap between communities (triangle).

3.2 Overlapping Normalized Cut

A popular definition of Ncut is the following [27]. For a graph $G = (V, E, w)$ with nodes partitioned into two disjoint sets, S and $S^c = V \setminus S$, the cut ∂S is defined as

$$\partial S = \{(a, b) \in E \colon a \in S, b \in S^c\}. \tag{1}$$

Then, the Ncut of partitioning S is defined as

$$\mathrm{Ncut}(S, S^c) = vol(\partial S)\left(\frac{1}{vol(S)} + \frac{1}{vol(S^c)}\right) \tag{2}$$

where

$$\begin{aligned} vol(\partial S) &= \sum_{i\in S, j\in S^c} W_{ij} \\ vol(S) &= \sum_{i\in S, j\in V} W_{ij} \\ vol(S^c) &= \sum_{i\in S^c, j\in V} W_{ij} \end{aligned} \tag{3}$$

in which W_{ij} is the weight of edge (i, j), $vol(\partial S)$ measures the volume of connections between S and S^c and $vol(S)$

measures the volume of connections between S and the rest of the graph.

Our proposed Overlapping Normalized Cut (ONcut) assesses the quality of a partitioning of edges into disjoint sets $S \subseteq E$ and $S^c = E \setminus S$. We define the cut ∂S as the set of nodes incident to at least one edge in both S and S^c:

$$\partial S = \left\{ i \in V : inc(i) \bigcap S \neq \emptyset, inc(i) \bigcap S^c \neq \emptyset \right\} \quad (4)$$

where $inc(i)$ is the set of edges incident to node i. The node set ∂S consists of what we called *bridge nodes* in Section 3.1. For each node $i \in \partial S$, we define its *dispersion* $\psi(i)$ as

$$\psi(i) = \frac{\left[\sum\limits_{e \in inc(i) \cap S} w(e) \right] \left[\sum\limits_{e \in inc(i) \cap S^c} w(e) \right]}{\sum\limits_{e \in inc(i)} w(e)}. \quad (5)$$

The dispersion of node i is a measure of the extent to which node i has an equal number of edges in S and S^c, and thus the extent to which it belongs to both communities. We define the volumes of the cut and of the edge sets by

$$\begin{aligned} vol(\partial S) &= \sum_{i \in \partial S} \psi(i) \\ vol(S) &= \sum_{e \in S} w(e) \\ vol(S^c) &= \sum_{e \in S^c} w(e). \end{aligned} \quad (6)$$

ONcut for an edge partitioning (S, S^c) is then defined as

$$\text{ONcut}(S, S^c) = vol(\partial S) \left(\frac{1}{vol(S)} + \frac{1}{vol(S^c)} \right). \quad (7)$$

From the above definitions, $vol(\partial S)$ is a measure of the number of nodes in common between S and S^c multiplied by their respective dispersions. This can be interpreted as the volume of the overlap between the community of nodes incident to at least one edge in S and the community of nodes incident to at least one edge in S^c. As $vol(S)$ is the total number of edges in S, it can be interpreted as the cohesiveness of the nodes incident to edges in S as measured by the total volume of interactions shared by these nodes. One may extract two edge communities (S, S^c) by minimizing ONcut, and overlapping node clusters result from these edge clusters. In the unweighted graph of Figure 1, $vol(S) = vol(S^c) = 13$, $vol(\partial S) = \frac{3}{2}$ and $\text{ONcut}(S, S^c) = \frac{3}{13}$. However, if we change the community membership of one of the edges connecting the central node to the cliques, we get $\text{ONcut}(S, S^c) = \frac{104}{315} > \frac{3}{13}$.

3.3 Vertex Reinforced Overlapping Normalized Cut

In some applications, one may have prior knowledge about the dispersion of nodes. For instance, if our aim is to detect overlapping communities of friends in an online social network, and if we know the areas of interest of each person, then people having a wide array of interests are likely to lie in the overlap between several communities of friends. Similarly, the number and the diversity of the groups in which a person is involved indicates the likelihood for this person to lie in some overlapping communities.

Even if no prior information is available, it is still possible to infer prior dispersions from the graph topology. As pointed out in [4], edges with high betweenness values are likely to connect nodes belonging to different communities. Following the same idea, nodes with high centrality values such as betweenness or closeness are likely to lie in the overlap between communities. Finally, if we have some preliminary partitioning of nodes obtained with any algorithm producing disjoint communities, we may assume that nodes that are not significantly closer to other nodes in the same community, compared to nodes in other communities, are more likely to lie in the overlap between different communities.

In all these cases, we define the *fidelity* $\phi(i)$ of a node i such that nodes with a low fidelity are likely to lie in several different communities while nodes with a high fidelity value tend to belong to a limited number of communities. Further details about how to choose the value of ϕ are given in Section 3.6.

The fidelities are included in our model as node weights. Defining S, ∂S and the node dispersion $\psi(i)$ in the same way as for ONcut, we redefine their respective volumes as

$$\begin{aligned} vol(\partial S) &= \sum_{i \in \partial S} \phi(i)\psi(i) \\ vol(S) &= \sum_{(i,j) \in S} W_{ij} \tfrac{\phi(i)+\phi(j)}{2} \\ vol(S^c) &= \sum_{(i,j) \in S^c} W_{ij} \tfrac{\phi(i)+\phi(j)}{2}. \end{aligned} \quad (8)$$

Then, the Vertex Reinforced Overlapping Normalized Cut (VRONcut) is defined in the same way as ONcut:

$$\text{VRONcut}(S, S^c) = vol(\partial S) \left(\frac{1}{vol(S)} + \frac{1}{vol(S^c)} \right). \quad (9)$$

ONcut is thus a particular case of VRONcut with $\phi(i) = 1$ for every node i. The definition of $vol(\partial S)$ gives more weight to an overlap consisting of nodes with high fidelities. In other words, for a given dispersion, the impact of a node on $vol(\partial S)$ will be greater if the node has a higher fidelity. Similarly, $vol(S)$ can be equivalently written as

$$vol(S) = \frac{1}{2} \sum_{i \in V} \phi(i) \sum_{e \in inc(i) \cap S} w(e). \quad (10)$$

The contribution of each node to $vol(S)$ is thus proportional to its fidelity. With such choices of $vol(\partial S)$ and $vol(S)$, a partitioning minimizing VRONcut favours nodes with a lower fidelity to lie in ∂S, which is what we are aiming at.

3.4 Spectral Clustering Algorithm for minimizing VRONcut

An advantage of the Ncut minimization is the existence of efficient spectral algorithms for finding an approximate solution to it [3]. In the biclustering case, these algorithms involve finding the smallest nonzero eigenvalue of the graph Laplacian and thresholding the entries of the corresponding eigenvector. The algorithm results from a relaxation of the Ncut problem. In this section, we show that a similar relaxation provides an efficient spectral clustering algorithm to minimize the VRONcut and produce communities.

For graph $G = (V, E, w, \phi)$ with edge weight vector w, node weight vector ϕ and incidence matrix M, we use the following notations: $d(i) = \sum\limits_{e \in E} M_{ie} w(e)$ is the degree of

node i and matrix D_v such that $(D_v)_{ij} = d(i)\delta_{ij}$ is the diagonal matrix of node degrees; $\delta(e) = w(e)\sum_{i\in V}\phi(i)M_{ie}$ is the degree of edge e and matrix D_e such that $(D_e)_{ef} = \delta(e)\delta_{ef}$ is the diagonal matrix of edge degrees[2]. Also, we define matrices $\Phi_{ij} = \phi(i)\delta_{ij}$, $\Omega_{ef} = w(e)\delta_{ef}$, and $A = \Omega M^T \Phi D_v^{-1} M\Omega$, where $\mathbf{1}$ denotes a vector of ones.

Our spectral algorithm is based on Theorem 1, whose proof is similar to the proof of the theorem presented in [27].

Theorem 1 (Reformulation of VRONcut minimization problem)**.** *For a graph $G = (V, E, w, \phi)$, define $S(z) = \{e \in E: z_e > 0\}$ for some vector $z \in \mathbb{R}^{|E|}$. Then, for*

$$[y^*, b^*, k^*] = \underset{y,k,b}{argmin} \frac{y^T(D_e - A)y}{y^T D_e y} \tag{11}$$

subject to the constraints of $y \in \{1, -b\}^{|E|}$, $b = \frac{k}{1-k}$ and $k = \frac{\sum_{e:y_e>0}\delta(e)}{\sum_e \delta(e)}$, a solution x^* to

$$\min_{x\in\{-1,1\}^{|E|}} VRONcut(S(x), S(x)^c) \tag{12}$$

is given by

$$x_e^* = \begin{cases} 1 & \text{if } y_e^* > 0 \\ -1 & \text{if } y_e^* \le 0 \end{cases} \tag{13}$$

for all $e \in E$. Moreover, $S(x^*) = S(y^*)$.

Proof. For $x \in \{-1, 1\}^{|E|}$, we define $k = \frac{\sum_{e:x_e>0}\delta(e)}{\sum_e \delta(e)}$. At the optimum, we necessarily have $0 < k < 1$, since $k = 0$ and $k = 1$ yield $\text{VRONcut}(S(x), S(x)^c) = \infty$. Then,

$$\frac{1}{vol(S(x))} + \frac{1}{vol(S(x)^c)} = \frac{2}{k\mathbf{1}^T D_e \mathbf{1}} + \frac{2}{(1-k)\mathbf{1}^T D_e \mathbf{1}}. \tag{14}$$

We also have

$$\begin{aligned} vol(\partial S(x)) &= \sum_{i\in\partial S(x)} \frac{\phi(i)}{d(i)} \sum_{\substack{e\in inc(i)\cap S(x) \\ f\in inc(i)\cap S^c(x)}} w(e)w(f) \\ &= \sum_{\substack{i\in V \\ e:x_e>0 \\ f:x_f<0}} \frac{\phi(i)}{d(i)} w(e)w(f)M_{ie}M_{if} \\ &= \frac{x^T}{2}(D_e - \Omega M^T \Phi D_v^{-1} M\Omega)\frac{x}{2}. \end{aligned} \tag{15}$$

Indeed, as $(D_e - \Omega M^T \Phi D_v^{-1} M\Omega)\mathbf{1} = 0$, we have

$$\begin{aligned} &\tfrac{x^T}{2}(D_e - \Omega M^T \Phi D_v^{-1} M\Omega)\tfrac{x}{2} \\ &= \tfrac{(x+\mathbf{1})^T}{2} D_e \tfrac{(x+\mathbf{1})}{2} - \tfrac{(x+\mathbf{1})^T}{2} \Omega M^T \Phi D_v^{-1} M\Omega \tfrac{(x+\mathbf{1})}{2} \\ &= \sum_{\substack{e:x(e)>0 \\ i\in V}} w(e)\phi(i)M_{ie} \frac{\sum_f w(f)M_{if} - \sum_{f:x(f)>0} w(f)M_{if}}{d(i)} \\ &= \sum_{\substack{i\in V \\ e:x_e>0 \\ f:x_f<0}} \frac{\phi(i)}{d(i)} w(e)w(f)M_{ie}M_{if}. \end{aligned} \tag{16}$$

Now, defining $\xi = \mathbf{1}^T D_e \mathbf{1}$, $A = \Omega M^T \Phi D_v^{-1} M\Omega$ and

$$\alpha(x) = \frac{x^T}{2}(D_e - A)\frac{x}{2} \tag{17}$$

then, from Equations (14) and (15), we obtain the objective function

$$\text{VRONcut}(S(x), S(x)^c) = \frac{2\alpha(x)}{\xi k(1-k)}. \tag{18}$$

2. Here, edge degree $\delta(e)$ should not be confused with the δ_{ef} function, which equals 1 if $e = f$ and 0 otherwise.

Defining $b = \frac{k}{1-k}$, the right member of Equation (18) becomes

$$\begin{aligned} \frac{\alpha(x)}{\xi k(1-k)} &= \frac{(1+b^2)\alpha(x)}{b\xi} + \frac{2b\alpha(x)}{b\xi} \\ &= \frac{1}{4}\left(\frac{(1+b^2)(x^T(D_e-A)x)}{b\mathbf{1}^T D_e \mathbf{1}} + \frac{2b(x^T(D_e-A)x)}{b\mathbf{1}^T D_e \mathbf{1}}\right) \\ &= \frac{1}{4b\mathbf{1}^T D_e \mathbf{1}}((\mathbf{1}+x) - b(\mathbf{1}-x))^T \\ &\quad (D_e - A)((\mathbf{1}+x) - b(\mathbf{1}-x)). \end{aligned} \tag{19}$$

Since $0 < k < 1$, we have $0 < b < \infty$ and we may define a new variable $y = h(x) := \frac{1}{2}((\mathbf{1}+x) - b(\mathbf{1}-x))$. We have $y_e \in \{1, -b\}$ and $\text{sign}(y_e) = \text{sign}(x_e)$ for all e. Moreover,

$$y^T D_e y = b(\sum_{y_e<0}\delta(e) + \sum_{y_e>0}\delta(e)) = b\mathbf{1}^T D_e \mathbf{1}. \tag{20}$$

Combining Equations (18), (19) and (20), and the definition of $y = h(x)$, and defining

$$F(y) = \frac{2y^T(D_e - A)y}{y^T D_e y}, \tag{21}$$

we have

$$\text{VRONcut}(S(x), S(x)^c) = F(y). \tag{22}$$

As a result, if one has a solution y^* to the problem

$$[y^*, b^*, k^*] = \underset{y,k,b}{\text{argmin}}\ F(y) \text{ s.t. } \begin{cases} y_e \in \{1, -b\} \\ b = \frac{k}{1-k} \\ k = \frac{\sum_{y_e>0}\delta(e)}{\sum\delta(e)}, \end{cases} \tag{23}$$

which corresponds to Problem (11), then, from Equation (22), a solution x^* to Problem (12) is $x^* = h^{-1}(y^*)$, namely

$$x_e^* = \begin{cases} 1 & \text{if } y_e^* > 0 \\ -1 & \text{if } y_e^* < 0. \end{cases} \tag{24}$$

Moreover, since $\text{sign}(x_e^*) = \text{sign}(y_e^*)$ for all e, we have

$$S(x^*) = S(y^*). \tag{25}$$

□

Hence, to minimize the VRONcut, we may find a solution y^* to Problem (11) and form the clusters $(S(y^*), S(y^*)^c)$ with $S(y^*) = \{e \in E : y_e > 0\}$. To provide a relaxation of Problem (11), we observe that any feasible solution y verifies

$$y^T D_e \mathbf{1} = (\sum_{y_e>0}\delta(e) - b\sum_{y_e<0}\delta(e)) = 0. \tag{26}$$

Keeping the orthogonality constraint expressed by Equation (26) and dropping the constraint $y_e \in \{1, -b\}$ making y discrete, a relaxed version of Problem (11) is given by

$$\min_y \frac{y^T(D_e - A)y}{y^T D_e y} \text{ subject to } y^T D_e \mathbf{1} = 0. \tag{27}$$

Introducing variable $z = D_e^{\frac{1}{2}} y$, the problem becomes

$$\min_z \frac{z^T \Delta z}{z^T z} \text{ subject to } z^T D_e^{\frac{1}{2}} \mathbf{1} = 0 \tag{28}$$

where

$$\Delta = D_e^{-\frac{1}{2}}(D_e - \Omega M^T \Phi D_v^{-1} M\Omega) D_e^{-\frac{1}{2}}. \tag{29}$$

The objective function of Problem (28) is the Rayleigh quotient of Δ. Moreover, if G is connected, then, Lemma

1 below[3] states that $D_e^{\frac{1}{2}}\mathbf{1}$ is the unique eigenvector of Δ associated with eigenvalue 0 (up to a multiplicative scalar). Hence, the orthogonality constraint (26) implies that, for a connected graph G, a solution z^* to Problem (28) is an eigenvector associated with the smallest nonzero eigenvalue of Δ.

Lemma 1.
For a graph $G = (V, E, w, \phi)$, with matrix Δ defined as in Equation (29), $D_e^{\frac{1}{2}}\mathbf{1}$ is an eigenvector of matrix Δ associated with eigenvalue 0. Moreover, if G is connected, then the multiplicity of 0 as an eigenvalue of Δ is 1.

Now, to solve Problem (28), our spectral clustering algorithm first computes matrix Δ. Then, it extracts its smallest nonzero eigenvalue and the corresponding eigenvector z^*, and classifies edges based on the signs of the entries of z^*. Finally, the belonging coefficient of a node i in community l is given by the proportion of edges incident to i that belong to edge community l. Algorithm 1, called *2 - Spectral Overlap Clustering* (2-SPOC), summarizes the process.

Algorithm 1 2-Spectral Overlap Clustering (2-SPOC)

Input: edge weight vector $w \in \mathbb{R}_+^{|E|}$, node weight vector $\phi \in \mathbb{R}_+^{|V|}$, incidence matrix $M \in \{0,1\}^{|V|\times|E|}$ of a connected graph.
Output: cluster fuzzy membership matrix C of nodes and cluster crisp membership vector σ of edges.
Parameter: threshold $\alpha \in [0, \frac{1}{2}]$.
Step 1: Compute edge and node degrees.
Step 2: Compute the smallest nonzero eigenvalue of the Laplacian Δ as in Equation (29) and its corresponding eigenvector u with $\|u\|_2 = 1$.
Step 3: Compute the cluster assignment of edges: $\sigma : E \to \{1, 2\}$, with

$$\sigma(e) = \begin{cases} 1 & \text{if } u_e > 0 \\ 2 & \text{otherwise.} \end{cases}$$

Step 4: Compute the node cluster assignment: $C \in [0,1]^{|V|\times 2}$ such that for each node i,

$$\begin{aligned} C_{i1} &= \tfrac{1}{d(u)} \textstyle\sum_{e:\sigma(e)=1} w(e) M_{ue} \\ C_{i2} &= 1 - C_{i1}. \end{aligned}$$

Step 5: For all nodes $i \in V$, if $C_{i1} < \alpha$, let

$$\begin{aligned} C_{i1} &\leftarrow 0 \\ C_{i2} &\leftarrow 1 \end{aligned}$$

and reversely if $C_{i2} < \alpha$.

We enforce $\|u\|_2 = 1$ since we are only interested in the signs of the entries of u. The algorithm requires a parameter $\alpha \in [0, \frac{1}{2}]$. If the belonging coefficient of a node to a community is below α, it is considered to be irrelevant and is set to 0. When both belonging coefficients of a node are above α, the output is a fuzzy community membership of the node. If one needs a binary membership, these two nonzero entries of the fuzzy membership vector are set to 1.

3. Lemma 1 can be proved by mimicking the proof of Proposition 2 in [3].

In practice, the Laplacian is not explicitly represented in our algorithm. Indeed, computing the smallest nonzero eigenvalue of the Laplacian Δ amounts to finding the second largest singular value and the corresponding left singular vector of

$$\Theta = D_e^{-\frac{1}{2}} \Omega M^T \Phi^{\frac{1}{2}} D_v^{-\frac{1}{2}} \tag{30}$$

which contains $O(|E|)$ nonzero elements. Hence, if the Lanczos algorithm [28] is applied to compute the singular value, our algorithm has the same time and space complexities as the node-based spectral clustering algorithms.

3.5 Hierarchical algorithm for K-clustering problem

In this section, we propose a hierarchical version of our algorithm to detect any number of clusters. To detect $K > 2$ clusters, we apply the 2-SPOC algorithm sequentially. Once two clusters are obtained, we reapply 2-SPOC to the subgraphs induced by both clusters. But, before proceeding, two issues must be addressed.

First, we show how the outputs of different calls of 2-SPOC are recombined. At each parent node in the tree, two communities of nodes are obtained, denoted by $C_{il} \in [0,1]$ for each node i and community $l \in \{1,2\}$ and by $\sigma_e \in \{1,2\}$ for each edge e. Then, the algorithm is run again on subgraphs $G^1 = (V^1, E^1)$ and $G^2 = (V^2, E^2)$, where $V^1 = \{i \in V : C_{i1} > 0\}$, $V^2 = \{i \in V : C_{i2} > 0\}$, $E^1 = \{e \in E : \sigma_e = 1\}$ and $E^2 = \{e \in E : \sigma_e = 2\}$.

After clustering both G^1 and G^2, we obtain two sets of fuzzy membership coefficients, C^1 and C^2, for nodes in V^1 and V^2, respectively, and two sets of crisp coefficients, σ^1 and σ^2, for edges in E^1 and E^2, respectively. These outputs define K_1 and K_2 communities, respectively. We concatenate these coefficients to obtain the final node memberships $\tilde{C}$ and edge memberships $\tilde{\sigma}$ for $K_1 + K_2$ communities:

$$\begin{aligned} \tilde{C}_{il} &= \begin{cases} C_{i1}C^1_{il} & \text{if } 1 \le l \le K_1 \\ C_{i2}C^2_{i(l-K_1)} & \text{if } K_1 + 1 \le l \le K_1 + K_2 \end{cases} \\ \tilde{\sigma}_e &= \begin{cases} \sigma^1_e & \text{if } e \in E_1 \\ \sigma^2_e + K_1 & \text{if } e \in E_2. \end{cases} \end{aligned} \tag{31}$$

Second, we introduce a stopping criterion for this hierarchical algorithm. If the number of communities is not known, then for a choice of parameter $\beta \in [0,1]$, the algorithm terminates (no further branching) when $\text{VRONcut}(S, S^c) > \beta$. This choice is justified by the fact that $\text{VRONcut}(S, S^c)$ lies in $[0,1]$. Moreover, as shown in experiments in Section 4, $\text{VRONcut}(S, S^c)$ does not vary with the number of nodes in the graph, but it varies with the relative volumes of the communities S and S^c, and of the cut ∂S. Hence, the choice of value for β should not depend on the absolute size of the graph. Experiments on real-world data (Section 5) show that values of β around 0.5 provide good results in practice. Algorithm 2 summarizes the above hierarchical process, referred to as the *Spectral Overlap Clustering* algorithm (SPOC).

In the particular case where the number of communities is known, we iteratively apply 2-SPOC to the subgraph corresponding to the cluster yielding the lowest value of VRONcut. The number of clusters is incremented by 1 at each step and the algorithm stops when the prescribed number of clusters is reached.

3.6 Choice of node weights

Recall that the fidelities or node weights should be chosen so that nodes with high weights are less likely to lie in the overlap between communities. When no prior knowledge is available, we introduce two heuristics to infer weights from the topology of the graph. The *betweenness* of node i is the fraction of shortest paths joining any two nodes of the graph that go through node i [29]. From Figure 1, bridge nodes tend to have larger betweenness values, since they must be present in paths connecting nodes that belong to different communities. The *closeness* of a node is the inverse of the average of its distances to all other nodes in the graph [29]. Nodes in the overlap should have higher closeness values as they are close to both communities.

Algorithm 2 Spectral Overlap Clustering (SPOC)

Input: edge weight vector $w \in \mathbb{R}_+^{|E|}$, node weight vector $\phi \in \mathbb{R}_+^{|V|}$, incidence matrix $M \in \{0,1\}^{|V|\times|E|}$ of a connected graph.
Output: cluster fuzzy memberships $\tilde{C}$ of nodes, cluster crisp memberships σ of edges and number K of clusters.
Parameters: threshold $\alpha \in [0, \frac{1}{2}]$, threshold $\beta \in [0,1]$.
Compute $[C, \sigma] \leftarrow$ 2-SPOC(w, ϕ, M, α).
Compute the value κ of VRONcut associated to σ.
if $\kappa > \beta$ **then**
  $C_{i1} = 1, \sigma(e) = 1, \forall i, e$.
  $K \leftarrow 1$.
else
  Extract (w^r, ϕ^r, M^r) for $r \in \{1,2\}$, the submatrices corresponding to the two communities defined by C, σ.
  $[C^1, \sigma^1, K_1] \leftarrow$ SPOC$(w^1, \phi^1, M^1, \alpha, \beta)$.
  $[C^2, \sigma^2, K_2] \leftarrow$ SPOC$(w^2, \phi^2, M^2, \alpha, \beta)$.
  $K \leftarrow K_1 + K_2$.
  Relabel clusters in σ_1 and σ_2 from 1 to K.
  Update σ by concatenating σ_1 and σ_2 and compute $\tilde{C}$:

$$\begin{aligned} \tilde{C}_{il} &= \begin{cases} C_{i1}C^1_{il} & \text{if } 1 \le l \le K_1 \\ C_{i2}C^2_{i(l-K_1)} & \text{if } K_1+1 \le l \le K_1+K_2 \end{cases} \\ \tilde{C}_{il} &\leftarrow 0 \text{ for each } i,l \text{ such that } \tilde{C}_{il} < \alpha. \end{aligned}$$

end if

The normalized betweenness and closeness provide coefficients η_i between 0 and 1 for each node i. For some parameter $\gamma > 1$, the fidelity ϕ_i of node i is defined as

$$\phi_i = \left(\frac{1-\eta_i}{1+\eta_i}\right)^{\gamma}. \tag{32}$$

We choose this function with a sharp initial decrease because nodes with $\eta_i \simeq 0$ are definitely not lying in the overlap between communities. However, as η_i becomes larger, this may indicate that either i is indeed in the overlap or i is a central node within its community. This justifies the choice of a rational function in the above form.

When detecting $K > 2$ communities, every time the graph is split into two, the betweenness or the closeness should be recomputed for both subgraphs. This is a time-consuming procedure due to the computation of shortest paths. Hence, we suggest computing all shortest paths once only for the entire graph G. Then, when splitting G into G^1 and G^2, we compute the corresponding coefficients η_i^1 and η_i^2 by extracting paths that lie entirely in G^1 or G^2, respectively. In doing so, we miss the shortest paths between nodes lying in the same community but that also include some edges from other communities along these paths. If the communities are well cohesive, such paths are unlikely to be observed. With this heuristic, shortest paths are computed only once for the whole graph. Experiments in Section 4 validate the approach.

3.7 Time complexity of the proposed algorithm

The computation of matrix Θ from Equation (30) in 2-SPOC requires $O(|V| + |E|)$ iterations. Computing the smallest nonzero eigenvalue of the Laplacian Δ is equivalent to finding the second largest singular value of matrix Θ with $O(|E|)$ nonzero elements. This can be performed by applying the Lanczos algorithm [28], which requires $O(l|E|)$ elementary operations, where l is the number of iterations of the Lanczos algorithm.

In the SPOC algorithm, at each level of the tree, 2-SPOC is applied r times respectively on matrices with $O(|E_1|)$,...,$O(|E_r|)$ nonzero elements, where r is the number of subgraphs at that level. Since $|E_1| + ... + |E_r| = |E|$, the total complexity at each level of the tree is $O(l|E|)$. As edges are further split into disjoint sets, there are $O(\log(|E|))$ (i.e. $O(\log(|V|))$ levels in the tree. Hence, the overall complexity of the SPOC algorithm is $O(l|E|\log(|V|))$.

Computing node closeness or betweenness requires $O(|V||E| + |V|^2\log(|V|))$ iterations [30]. Hence, the overall complexity is $O(l|E|\log(|V|) + |E||V| + |V|^2\log(|V|))$. In applications involving sparse graphs, one has $|E| = O(|V|)$, so the complexity reduces to $O(l|V|^2\log(|V|))$ with node weights and $O(l|V|\log(|V|))$ without node weights, making our approach more efficient than other methods such as the clique percolation (see Section 2). The number l of iterations for the convergence of the Lanczos algorithm depends on the condition number of the corresponding singular vector, but it is generally small compared to the number of nonzero elements in the matrix [28], [31]. As described in Section 4, we ran 2-SPOC without node weights on a graph of 10^6 nodes and 10^7 edges in one minute with a `Matlab` program on a standard personal computer.

4 Tests on synthetic data

We first define a quality measure for overlapping communities. Then, we show how to build synthetic graphs with overlapping communities, and we analyze the performance of our algorithm in different situations. Finally, we compare our algorithm to six other methods of overlapping community detection.

4.1 Measuring the cluster quality

The Normalized Mutual Information (NMI) [32] is a standard measure for the quality of a partitioning of nodes into non-overlapping clusters. An extension of the NMI for overlapping clusters was proposed in [9]. We refer to it as Overlap-Normalized Mutual Information (ONMI). The true and the predicted clusters are denoted by sets C_l for

$1 \leq l \leq K$ and $\tilde{C}_l$ for $1 \leq l \leq \tilde{K}$, where $\tilde{K}$ is the predicted number of clusters. We introduce two random variables:

$$\begin{aligned} X_l &= 1 \text{ denotes the event that a node belongs to } C_l \\ \tilde{X}_l &= 1 \text{ denotes the event that a node belongs to } \tilde{C}_l. \end{aligned} \tag{33}$$

The ONMI is then given by

$$\text{ONMI}(\mathbf{X}, \tilde{\mathbf{X}}) = 1 - \frac{1}{2}(H(\mathbf{X}|\tilde{\mathbf{X}}) + H(\tilde{\mathbf{X}}|\mathbf{X})) \tag{34}$$

where the normalized entropy of all X_k given all $\tilde{X}_l$ is

$$H(\mathbf{X}|\tilde{\mathbf{X}}) = \frac{1}{K}\sum_k \frac{H(X_k|\tilde{\mathbf{X}})}{H(X_k)}. \tag{35}$$

We refer to [9] for further details about the computation of the conditional entropy $H(X_k|\tilde{\mathbf{X}})$ of each cluster. When $\tilde{\mathbf{X}}$ is independent of $\mathbf{X}$, $\text{ONMI}(\mathbf{X}, \tilde{\mathbf{X}})$ is 0. Hence, any value of the ONMI that is larger than 0 is an improvement compared to random cluster assignments. It is noted that the ONMI cannot be used to measure the quality of fuzzy community memberships. Hence, in our SPOC algorithm, we use parameter α to threshold the fuzzy memberships so as to produce binary community membership vectors.

4.2 Synthetic graphs with overlapping communities

We now propose a random graph model to build a graph with overlapping communities. This model is related to the standard stochastic blockmodel [19]. We refer to it as *Overlapping Stochastic Blockmodel* (OSB). This model can also be viewed as a generalization of the Planted L-Partition model [33].

Definition 1 ($\text{OSB}(v, P, \zeta, \theta, K)$)**.** *An unweighted graph $G = (V, E)$ is generated by an Overlapping Stochastic Blockmodel with parameters $K \in \mathbb{N}$, $P \in [0,1]^{K\times K}$, $\rho \in [0,1]^K$ satisfying $\sum_l \rho_l = 1$, $\zeta > 0$ and $\theta \in [0,1]$, if*

1) *for each node $i \in V$,*
   - *a number k_i of communities is drawn from distribution $P\{k_i = k\} = \frac{\theta(1-\theta)^{k-1}}{\sum_{l=0}^{K-1}\theta(1-\theta)^l}$,*
   - *k_i communities are drawn without repetition from the categorical distribution defined by vector ρ,*
   - *a community distribution for node i is drawn from a Dirichlet distribution $\pi^i \sim Dir(\zeta \mathbf{1}_{k_i})$, where $\mathbf{1}_{k_i}$ is the k_i-dimensional vector of ones,*
2) *for each pair of nodes $i, j \in V$, we have*

$$P\{(i,j) \in E\} = \sum_{kl} \pi_k^i \pi_l^j P_{kl}. \tag{36}$$

To produce an undirected graph, we choose P to be symmetric and then symmetrize the resulting graph by making each directed edge bidirectional.

The distribution of the number of communities per node ensures exponential decrease in the probability of picking a large number of communities, which is realistic as only a small fraction of nodes are expected to belong to several communities. This distribution is controlled by parameter θ. A draw of communities from the categorical distribution induces a variation in the sizes of the communities, where a community with a large corresponding entry of vector ρ contains a relatively large number of nodes. The degree of membership of a node in each community is modelled as a vector of probabilities with k_i entries, where k_i is the number of communities for node i. We draw this vector of community membership from a Dirichlet distribution since this distribution naturally produces probability vectors π^i such that, when increasing the concentration parameter ζ, the entries of π^i become closer to uniform. Finally, since the probability of an edge (i, j), given that i is in community k and j is in community l, is a parameter P_{kl}, we have that the probability of edge (i, j) is the sum over all pairs k, l of P_{kl} multiplied by membership probabilities π_k^i and π_l^j.

The individual contributions of the different parameters can be described in the following way. Parameter K governs the overall number of communities. Parameter θ governs the number of communities for each node: the lower θ, the larger the number of nodes that are members of a large number of communities. Parameter ρ governs the relative size of each community: the larger ρ_l, the higher the number of nodes contained in community l with respect to other communities. Parameter ζ governs the balance of contributions of each community to each node: for a node belonging to more than two communities, a small value of ζ yields unbalanced contributions from each community, with one community generating a large proportion of the edges incident to that node. Parameter P_{kl} governs the probability of having an edge connecting two nodes belonging to communities k and l, respectively. To generate a graph with cohesive communities, we choose $P_{kk} >> P_{kl}$ for all k, l.

Next, we run two types of tests. First, we measure the sensitivity of SPOC to the *overlap setting* by fixing the values of the parameters of SPOC and letting the value of one parameter of OSB vary in each test. Then, we fix all parameters of OSB and let each parameter of SPOC vary.

4.3 Sensitivity of SPOC to the overlap setting

In this test, we choose the following values for the parameters of our SPOC algorithm: $\alpha = 0.2$ as this value provides good results in general; $K = 3$ (we set the number of clusters to allow focusing on the task of community detection); as K is given, parameter β is not necessary. We do not include node weights. The impact of node weights is analyzed in Section 4.4. In what follows, the effect of different parameters of OSB are analysed. In each test, we let one parameter of OSB vary while the values of other parameters are set to the following standard values: $n = 1000$ nodes, $K = 3$ communities, $\theta = 0.7$ (which results, for instance, in approximately 70% of nodes belonging to a single community, 20% belonging to two communities and 10% belonging to three communities), $\rho = [1/K, ..., 1/K]$ (balanced communities), $\zeta = 3$ (which typically results in distributions of the form $[0.4, 0.3, 0.3]$ for three communities), $P_{kk} = 0.5$ and $P_{kl} = 0.01$ for $k \neq l$. The graphs in Figure 2 depict the evolution of the ONMI as a function of each parameter of the OSB graph model respectively.

Effect of θ (number of nodes with multiple communities): we let θ vary linearly between 0.1 and 1. As expected, the ONMI decreases when θ decreases (Figure 2(a)). However, even with $\theta = 0.4$, the ONMI stays above 0.6. Moreover, for θ above 0.7, the ONMI is above 0.8.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2892096, IEEE Transactions on Knowledge and Data Engineering

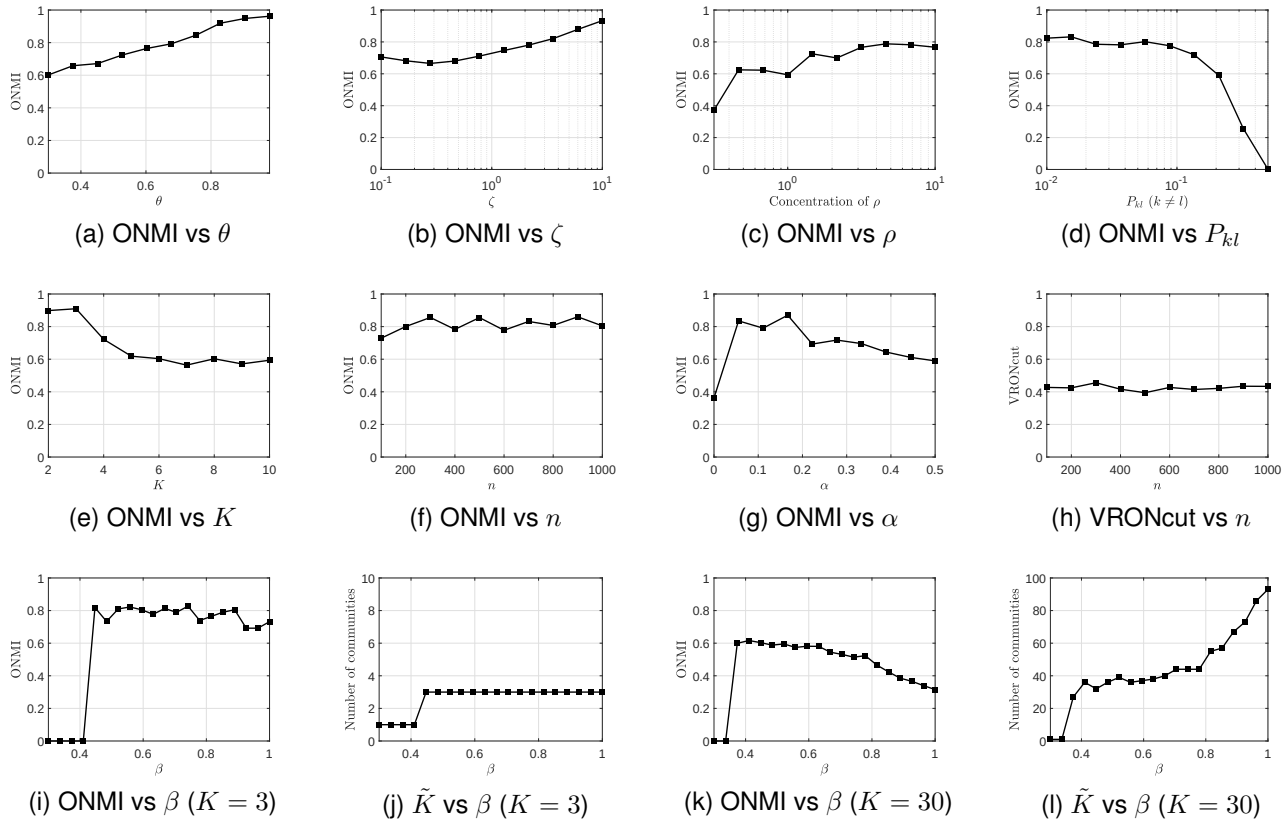


Figure 2. Performance plots (details in the text).

Hence, the quality of the output is not highly sensitive to the number of communities assigned to each node.

Effect of ζ (distribution over communities): we let ζ vary logarithmically between 0.1 and 10. $\zeta = 0.1$ yields an unbalanced distribution over clusters, for instance $[0.67, 0.2, 0.13]$ for three clusters, while $\zeta = 10$ yields a balanced distribution. In Figure 2(b), we observe that the ONMI increase with ζ. Indeed, when ζ is low, a given node may have a very low belonging factor to some community. Thus, the resulting number of edges is also small, and the membership of this node to that community can hardly be detected by our SPOC algorithm. Nevertheless, the ONMI stays above 0.6 for this range of ζ values.

Effect of ρ (relative sizes of communities): vector ρ commands the relative number of nodes in each community. We draw ρ from a symmetric Dirichlet distribution [34] with concentration parameter varying logarithmically between 0.3 and 10. For three communities, a concentration parameter of 0.3 yields, for instance, $\rho = [0.1, 0.05, 0.85]$ (unbalanced communities). A concentration parameter of 10 typically yields a ρ close to $[1/3, 1/3, 1/3]$. From Figure 2(c), we observe that 0.5 is the critical value for the concentration parameter of ρ: above 0.5, communities are balanced enough and the ONMI is above 0.6; below that value, communities become highly unbalanced and the ONMI drops at 0.4. Hence, our SPOC algorithm is not able to detect highly unbalanced communities. However, this issue is still largely an open problem, leaving room for future improvements.

Effect of P (probability of edges between communities): we set $P_{kk} = 0.5$ and let P_{kl} $(k \neq l)$ vary logarithmically between 0.01 and 0.5. The resulting graph is displayed in Figure 2(d). As expected, the quality of the output decreases as P_{kl} $(k \neq l)$ gets closer to 0.5, and the ONMI is 0 when $P_{kl} = 0.5$ for all k, l. In this case, the probability of connection within communities is not higher than the probability of connection between communities. A critical value for P_{kl} is around 0.2. Below that value, the ONMI is above 0.6 and stays approximately constant. Above that value, the ONMI drops dramatically. A value of $P_{kl} > 0$, for $k \neq l$, is interpreted as allowing some edges to lie in the overlap between communities. In this way, we measure the efficiency of our algorithm when both edges and nodes belong to more than one community. From the above experiment, we observe that, with a slight amount of overlap of edge communities (typically, $P_{kl} \leq 0.4 P_{kk}$ for $k \neq l$), this overlap does not have a notable impact on the output of SPOC. The concern of having overlapping communities of both nodes and edges was raised in [1], and, to the best of our knowledge, there is no other research testing clustering algorithms in that setting.

Effect of the number K of clusters: Figure 2(e) shows the evolution of the ONMI with a growing number of clusters. As expected, the quality of the result decreases slightly when the number K of clusters grows. This is due to the fact that the hierarchical framework of SPOC might introduce additional errors when K grows. However, the ONMI stays above 0.6 and it stabilizes around that value for $K \geq 5$.

Effect of the number of nodes: with a number n of nodes varying from 100 to 1000, Figure 2(f) shows that the

impact of n is limited, although the ONMI seems to slightly increase with the number of nodes. This was expected as SPOC is sensitive to the relative sizes of the communities (parameter ρ), but not to their absolute sizes.

Computational time: we measure the evolution of the computational time for a number of nodes varying between 10^3 and 10^6. We restrict the number of edges to ten times the number of nodes, which yields an average degree of 10 and is coherent with the degrees encountered in real-world networks. Table 1 displays the average computational times for 100 experiments, using a `Matlab` implementation of 2-SPOC with $l = 1000$ iterations for Lanczos algorithm, on a standard computer with a 2.5 GHz processor and 8.0 GB memory. Even for 10^6 nodes, the computational time is about 60 seconds, proving the efficiency of our algorithm. With further improvement in the implementation, it should be able to process large social networks with tens of millions of nodes within seconds. In contrast, some other methods such as the clique percolation [6] were shown to be intractable for such networks.

Number of nodes	Number of edges	Time (seconds)
10^3	10^4	0.1068
10^4	10^5	1.1556
10^5	10^6	8.8261
10^6	10^7	65.8455

Table 1
Computational time for different numbers of nodes and edges.

We draw the following three conclusions. First, our SPOC algorithm is efficient regardless of the number of communities per node and the belonging coefficient of each node to each community. Second, for the communities to be detected, the concentration parameter of the relative size of each community must be above 0.5 and the probability of connections between communities must satisfy $P_{kl} << P_{kk}$ for $k \neq l$ (typically, $P_{kl} \leq 0.4 P_{kk}$). Third, the SPOC algorithm is efficient regardless of the number of nodes and the number K of communities, and it runs in a reasonable time for large networks containing millions of nodes.

4.4 Impact of parameters of the SPOC algorithm

In this section, we set the values of the parameters of OSB to the standard values given in Section 4.3 and we investigate the impact of the parameters of the SPOC algorithm.

Impact of α: recall that α is a threshold in the SPOC algorithm in order to filter out low and irrelevant community memberships. Figure 2(g) shows the evolution of the ONMI with α varying between 0 and 0.5. The ONMI is the lowest at $\alpha = 0$, which is due to a poor precision, namely spurious community memberships that must be filtered out. The value of the ONMI is maximal for α between 0.05 and 0.2. Then, it decreases due to a poor recall, because our SPOC algorithm fails to detect all cluster memberships when α is too high. It is thus reasonable to choose a small nonzero value for α (typically, 0.2).

Variation of VRONcut as a function of n: to check the validity of the stopping criterion in our hierarchical framework, we measure the variation of VRONcut as a function of the number n of nodes for OSB graphs, keeping other parameters of the model unchanged. As depicted by Figure 2(h), VRONcut does not vary significantly with n. Instead, it only depends on the relative volumes of cuts and communities, as we intended.

Impact of β: recall that, when the number K of communities is not known, the algorithm stops branching whenever the VRONcut occurring at a given node of the tree exceeds β. We test the performance of our SPOC algorithm on an OSB graph with standard parameter values as given above, but without providing the number of communities. We also keep $\alpha = 0.2$. Figures 2(i), 2(j), 2(k) and 2(l) display the ONMI and the detected number $\tilde{K}$ of communities for a true number of communities of $K = 3$ and $K = 30$, respectively. For the chosen values of β and K, the average performance for 100 different tests is displayed. In both cases, the ONMI reaches a maximal value in an interval centered at 0.5 (approximately, $[0.4, 0.6]$). The number of communities is well detected when $\beta > 0.4$ for $K = 3$ and it is approximately well detected when $0.4 \leq \beta \leq 0.8$ for $K = 30$. Hence, our algorithm is not highly sensitive to the value of parameter β. While some tuning might improve the quality of the result, $\beta = 0.5$ is a reasonable choice.

Impact of node weights: we test the performance of our SPOC algorithm when including prior node weights or *fidelities*. We test two choices for the weights, based on the node betweenness and the node closeness, respectively, and with parameter $\gamma = 2$ in Equation (32). Figure 3 shows the evolution of the ONMI as a function of parameter θ for all three settings. The quality is slightly improved when including either closeness or betweenness-based fidelities, with closeness yielding the best result.

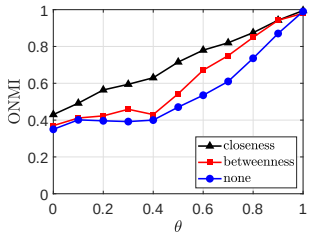


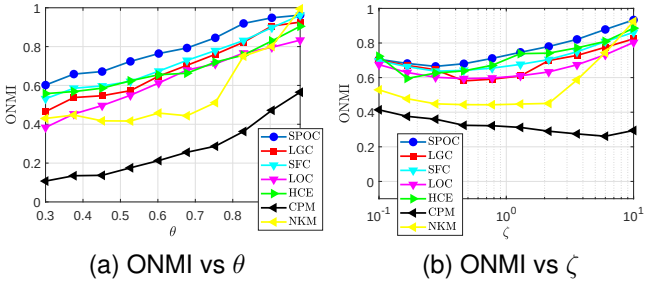
Figure 3. Evolution of the ONMI as a function of parameter θ of an OSB graph model of 1000 nodes and $K = 5$ blocks for SPOC with unit, betweenness-based and closeness-based node weights respectively.

4.5 Comparison with other methods

We compare SPOC with six other algorithms reviewed in Section 2: (1) the *Clique Percolation Method* (CPM) [7], with parameter $k = 3$ for the k-cliques, as suggested in [7]; (2) an algorithm we refer to as *Hierarchical Clustering of Edges* (HCE) [13], which applies agglomerative clustering with single linkage to cluster edges and extracts overlapping node communities from edge clusters, with the number of communities as a parameter; (3) an algorithm we refer to as *Line Graph Clustering* (LGC) [14], which applies a non-parametric modularity-based algorithm to the line graph to cluster edges, and then extracts overlapping clusters of nodes from edge clusters; (4) an algorithm we call *Spectral Fuzzy Clustering* (SFC) [16], which extends the traditional

spectral clustering algorithm, and applies fuzzy c-means instead of k-means to the entries of the principal eigenvectors of the Laplacian (the parameters are the number K of communities, the fuzzy partition exponent m which is set to 2, and the threshold for crisp communities λ which is set to 0.1, as in [16]); (5) the NEO-k-means (NKM) algorithm [18], which is a kernelized version of the k-means algorithm for the extraction of overlapping communities in graphs (the number of communities is a parameter, the communities are initialized with spectral clustering, parameter β commanding the number of unassigned nodes is set to 0, and parameter α commanding the average number of communities per node is determined by a search algorithm provided in [18]); (6) a method we call *Local Fitness Optimization* (LOC) [9], which is based on local improvement of the community fitness measuring the ratio between the numbers of inner and of outer connections of each cluster. The resolution parameter for LOC is set to 1, as suggested when no prior knowledge is available, and the seed nodes are generated randomly.

For CPM, the worst case time complexity is exponential [6]. HCE and LGC have a polynomial complexity but they require an explicit representation of an edge graph, which is costly on dense graphs. SFC has a time complexity of $O(l|E| + mK^2|V|)$ for a number l of iterations in the computation of eigenvalues and a number m of iterations in the fuzzy c-means algorithm [16]. NKM has a time complexity of $O(mK|V|\log(|V|))$ with m being the total number of iterations of the process [18]. LOC has a time complexity of $O(|V|^2)$ [9].

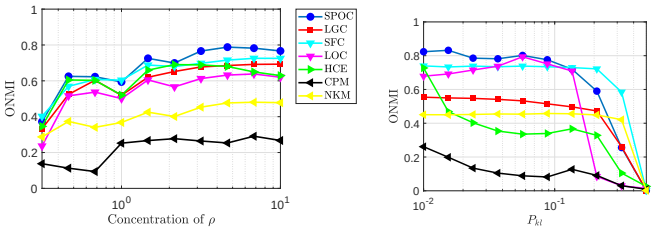


(a) ONMI vs θ (b) ONMI vs ζ

Figure 4. ONMI as a function of θ (left) and ζ (right) of the OSB graph model for SPOC, LGC, SFC, LOC, HCE, NKM and CPM algorithms.

Figure 4(a) shows the evolution of the ONMI as a function of parameter θ of the OSB graph model for all six algorithms, while the values of the other parameters of OSB and SPOC are maintained as before. Our SPOC algorithm yields the highest ONMI regardless of θ. The performances of HCE and LGC are similar to that of our algorithm since they also rely on edge clustering. The superiority of our method is due to the use of a cut-based approach instead of a modularity-based approach (LGC), or an agglomerative clustering of edges (HCE), which induces significant errors when the number of nodes in the overlap is large (low θ). SFC produces a lower ONMI than SPOC since there is no clear theoretical justification in applying the fuzzy c-means algorithm to the principal eigenvectors of the graph Laplacian. LOC also achieves a low ONMI due to the impact of random seed nodes. NKM fails when the number of nodes in the overlap is large since its ONMI is below 0.6 for θ up to 0.8. This is due to the difficulty of setting the number of nodes in the overlap as a parameter. Finally, the poor performance of CPM is due to the presence of sparsely connected communities when $P_{kk} = 0.5$, which causes a clique-based approach to fail.

Figure 4(b) shows the evolution of the ONMI as a function of ζ. Again, our SPOC algorithm achieves the highest performance regardless of ζ while LGC, SFC, LOC and HCE produce a slightly lower ONMI. NKM yields a poor performance for values of ζ below 1.3. We recall that ζ is the concentration of the community belonging coefficients for each node. The ONMI increases with ζ for all algorithms except for CPM. Indeed, a high value of ζ results in nodes having an approximately balanced number of edges in each community they are members of, implying a higher risk for two different communities to share cliques.



(a) ONMI vs ρ (b) ONMI vs P_{kl}

Figure 5. ONMI as a function of ρ (left) and P_{kl} for $k \neq l$ (right) of the OSB graph model for SPOC and other six algorithms.

Figure 5(a) shows the evolution of the ONMI as a function of the concentration of the probability vector ρ of community memberships of nodes. All algorithms perform poorly when the communities are unbalanced and their ONMI increases with the concentration of ρ. Our SPOC algorithm produces the highest ONMI (or close to the highest) regardless of ρ. SFC, LGC and HCE, which share similarities with our approach, also achieve high values of the ONMI, and so does LOC. In contrast, NKM and CPM achieve lower values of ONMI (typically below 0.4) when the communities are unbalanced. Indeed, NKM is a generalization of the node-based Ncut which assumes the presence of balanced communities. As for CPM, its inability to detect sparsely connected communities is slightly more prominent when the communities have unbalanced sizes.

Finally, Figure 5(b) shows the evolution of the ONMI as a function of the probability of connection between clusters. While our SPOC algorithm outperforms other methods for a probability P_{kl} below 0.2, it is outperformed by SFC for $P_{kl} = 0.2$ and by SFC and NKM for $P_{kl} = 0.3$, which perform slightly better in the presence of a large number of inter-cluster edges. As expected, our method essentially succeeds in the detection of overlapping communities of nodes for which the associated edge communities are disjoint. Moreover, our SPOC algorithm outperforms LGC, LOC, HCE and CPM for each value of P_{kl}.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2892096, IEEE Transactions on Knowledge and Data Engineering

5 Application to Real-World Data

We apply our SPOC algorithm to five real-world networks: Zachary's karate network, Word Association network, and networks based on Amazon, DBLP and YouTube.

5.1 Zachary's karate network

Zachary's network [35] is a social network depicting 78 friendship ties among 34 members of a karate club (Figure 6). A conflict between the instructor (Node 1) and the administrator (Node 34) partitioned the network into two communities. As shown in [14], some nodes lie close to the boundary between both ground-truth communities. This is a reason for seeking overlapping communities in the network.

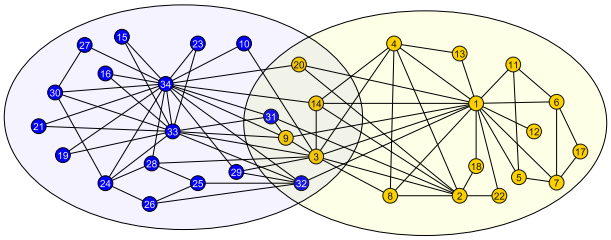


Figure 6. Zachary's karate club network: Colours of nodes denote how members were partitioned after the conflict. The two communities discovered by our SPOC algorithm are circled.

We apply the SPOC algorithm for detecting two communities with parameter $\alpha = 0$ since there is no need to filter out low community memberships on such a small network. The result is displayed in Figure 6. Four of the six nodes in the overlap have direct interactions with nodes 1 and 34, and the other two nodes are connected to multiple neighbours of 1 and 34, which is a sign that the output of SPOC is relevant. In Table 2, we also display the nodes contained in the overlap and the ONMI with respect to the true non-overlapping communities when applying SPOC and the six comparison methods described in Section 4.5. We expect the extracted overlapping communities to be coherent with the ground-truth communities and hence to produce a high ONMI. The number $K = 2$ of communities is provided as a parameter for SFC, HCE and NKM, and all other parameters are obtained as in Section 4.5. First, we observe that SPOC and LGC produce the same results. SFC fails to detect nodes 9 and 32 (both connected to nodes 34 and 1) as being part of the overlap. LOC fails to include nodes 14, 20 and 32 in the overlap although they are connected to both nodes 1 and 34. NKM assigns node 29 to both communities while it clearly has a stronger connection to node 34 and the left community. Finally, both HCE and CPM produce the lowest ONMI value and they fail to identify any meaningful overlap between the communities, which is partly due to the sparsity of the graph in the case of CPM. Overall, SPOC and LGC both succeed in identifying a significant and relevant overlap between the communities while being coherent with the ground-truth communities (high ONMI).

Algorithm	ONMI	Nodes in overlap
SPOC	0.655	3, 9, 14, 20, 31, 32
LGC	0.655	3, 9, 14, 20, 31, 32
SFC	0.708	3, 14, 20
LOC	0.33	3, 9, 10, 31
HCE	0.0957	1
CPM	0.167	1
NKM	0.613	3, 9, 10, 14, 20, 29, 31

Table 2
Performances of our SPOC and other six algorithms on Zachary's karate network.

5.2 Word Association

Word Association network [36] is a lexical network depicting relationships of association between English words as perceived by a group of individuals. The network forms an unweighted graph of $23,219$ nodes (words) and $325,624$ edges (relationships). As suggested in [13], we use the four quantity described below to measure the quality of the partitioning. Unlike [13], we use Google's pretrained word embeddings [37] instead of WordNet [38] to compute semantic similarities between words, since the coverage of word embeddings is higher. The cosine similarity between embeddings is used as a word similarity measure [37].

The *Community Quality* is given by the average similarity of words lying in the same community divided by the average similarity of all words in the thesaurus. The *Overlap Quality* is the mutual information between the number of communities of each node and its average similarity with all other words, which is interpreted as a measure of the probability for each node to lie in the overlap between communities. The *Community Coverage* measures the fraction of nodes lying in at least one community of minimum three nodes. The *Overlap Coverage* is given by the average number of meaningful communities per node, where a meaningful community is a community containing at least three nodes.

We choose the following values for parameters of our SPOC algorithm: $\alpha = 0.2$ as before, $\beta = 0.5$, and closeness-based node weights. SPOC detects 458 communities. We first analyze the histograms of the number of nodes per community (Figure 7 (left)) and the number of communities per node (Figure 7 (right)). The average community size is 130. Only one community contains less than three nodes. The largest community contains 1363 nodes, which shows the ability of our algorithm to detect communities of different sizes. The distribution of the number of communities per node follows a power law. We interpret the number of communities as the "number of meanings" of a word or the number of synsets in which a word finds itself. In a similar manner, it was also observed that the degree distribution of the WordNet graph follows a power law [39].

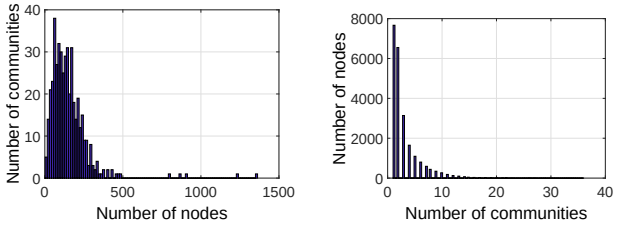


Figure 7. Histogram of the number of nodes per community (left) and histogram of the number of communities per node (right).

Next, we analyze the performance of SPOC in terms of the four quality measures described above. We also run LGC, SFC, LOC, HCE and NKM (but not CPM due to its high computational cost on large datasets). The number K of communities is set to 500 for SFC and NKM, which approximately corresponds to the number of communities discovered by the SPOC and LOC. The threshold parameter for HCE is set to 0.5 and the values of all other parameters are obtained as in Section 4.5. Table 3 summarizes the performances of these algorithms (with the overlap quality of each method divided by the overlap quality achieved by SPOC). SPOC outperforms all others in terms of community quality, showing that its output communities are coherent with the semantic information expressed by word embeddings. Our algorithm also achieves the highest overlap quality. Except for LGC and HCE, all methods achieve a community coverage close to 100%. Finally, HCE assigns each node to 5.235 communities on average, which is slightly more than the 4.404 communities discovered by our SPOC algorithm.

	Community Quality	Overlap Quality	Community Coverage	Overlap Coverage
SPOC	**1.531**	**1.00**	**1.00**	4.404
LGC	1.304	0.319	0.818	1.067
SFC	1.056	0.162	0.999	1.046
LOC	1.375	0.627	**1.00**	4.474
HCE	1.042	0.325	0.924	**5.235**
NKM	1.139	0.112	**1.00**	1.023

Table 3
Performances of our SPOC algorithm and that of LGC, SFC, LOC, HCE and NKM on the Word Association network.

5.3 Three large networks: Amazon, DBLP and YouTube

Finally, we apply our algorithm to three large real-world networks with overlapping ground-truth communities: Amazon, DBLP and YouTube datasets [40]. Amazon network is a co-purchasing network of products with $334,863$ nodes and $925,872$ edges. DBLP dataset is a social network depicting relationships of co-authorship between authors of scientific papers containing $317,080$ nodes and $1,049,866$ edges. The third dataset is extracted from the social network formed by YouTube users. It consists of $1,134,890$ users and $2,987,624$ friendship ties. All graphs are connected. Since we expect the presence of noisy edges which do not contribute to the community structure of the graphs, we set the parameters of our SPOC algorithm to 0.3 for α. We keep $\beta = 0.5$, and node weights equal to 1 to reduce the time complexity. We also test LGC, SFC, LOC and NKM algorithms. The number K of communities is set to $25,000$ for SFC and NKM, as suggested in [18], and the proportion of unassigned nodes for NKM is set to 0.0001 as in [18]. The values of all other parameters are obtained as in Section 4.5. We do not test neither CPM nor HCE since these algorithms do not scale to networks of such sizes.

Ground-truth overlapping communities are available for all three networks [40]. Table 4 displays the ONMI scores achieved by all five methods. Our SPOC algorithm outperforms the second best performing approach, the LOC algorithm, by 22, 21 and 29% of ONMI on Amazon, DBLP and YouTube datasets, respectively.

	SPOC	LGC	SFC	LOC	NKM
Amazon	**0.2558**	0.1422	0.0875	0.2093	0.1540
DBLP	**0.2069**	0.1336	0.1222	0.1717	0.1383
YouTube	**0.1879**	0.0919	0.1171	0.1449	0.1310

Table 4
ONMI achieved by our SPOC algorithm and LGC, SFC, LOC and NKM on Amazon, DBLP and YouTube datasets.

6 Conclusion and Perspectives

We proposed a new method for finding overlapping communities in large-scale networks. This method is based on an extension of Normalized Cut. We introduced a hierarchical version of the algorithm to detect the number of clusters, and we included node weights expressing the likelihood for any node to lie in the overlap between communities. These node weights can be inferred from metadata or from node centrality measures such as the node closeness. We demonstrated that our method significantly outperforms the state-of-the-art algorithms on both synthetic and real-world data. We verified the scalability of our method by processing large graphs with millions of nodes and edges in about a minute using `Matlab` on a regular personal computer.

As a future research direction, we will investigate the choice of values for the tolerance parameter β in the hierarchical framework. Since we did not fully investigate the impact of node weights, we would test our algorithm on real-world datasets in which a prior likelihood for each node to be part of several communities is available. We did not test our algorithm on weighted graphs yet, which will be done in the near future. Finally, the scalability of our algorithm suggests that we could apply it to very large social networks such as large-scale Facebook-based networks. To that end, we will try to perform some operations of the new algorithm in parallel on super-sized datasets.

Acknowledgements

This research was supported by the Hong Kong Research Grants Council under the GRF Grant CityU11200317.

References

[1] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75 – 174, 2010.
[2] R. Sibson, "SLINK: An optimally efficient algorithm for the single-link cluster method," *The Computer Journal*, vol. 16, no. 1, pp. 30–34, 1973.
[3] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.
[4] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
[5] J. Reichardt and D. R. White, "Role models for complex networks," *The European Physical Journal B*, vol. 60, no. 2, pp. 217–224, 2007.
[6] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: The state-of-the-art and comparative study," *ACM Computing Surveys*, vol. 45, no. 4, pp. 43:1–43:35, 2013.
[7] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, pp. 814–818, 2005.
[8] J. Baumes, M. K. Goldberg, M. S. Krishnamoorthy, M. Magdon-Ismail, and N. Preston, "Finding communities by clustering a graph into overlapping subgraphs," in *Proc. of the IADIS International Conference on Applied Computing (IADIS 2005)*, 2005, pp. 97–104.

[9] A. Lancichinetti, S. Fortunato, and J. Kertsz, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, vol. 11, no. 3, p. 033015, 2009.
[10] P. Latouche, E. Birmele, and C. Ambroise, "Overlapping stochastic block models with application to the french political blogosphere," *The Annals of Applied Statistics*, vol. 5, no. 1, pp. 309–336, 2011.
[11] T. Nepusz, A. Petróczi, L. Négyessy, and F. Bazsó, "Fuzzy communities and the concept of bridgeness in complex networks," *Physical Review E*, vol. 77, p. 016107, 2008.
[12] J. Yang and J. Leskovec, "Overlapping community detection at scale: A nonnegative matrix factorization approach," in *Proc. of the 6th ACM International Conference on Web Search and Data Mining (WSDM 2013)*, 2013, pp. 587–596.
[13] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *nature*, vol. 466, pp. 761–764, 2010.
[14] T. S. Evans and R. Lambiotte, "Line graphs of weighted networks for overlapping communities," *The European Physical Journal B*, vol. 77, no. 2, pp. 265–272, 2010.
[15] X. Deng, G. Li, M. Dong, and K. Ota, "Finding overlapping communities based on markov chain and link clustering," *Peer-to-Peer Networking and Applications*, vol. 10, no. 2, pp. 411–420, 2017.
[16] S. Zhang, R.-S. Wang, and X.-S. Zhang, "Identification of overlapping community structure in complex networks using fuzzy c-means clustering," *Physica A: Statistical Mechanics and its Applications*, vol. 374, no. 1, pp. 483 – 490, 2007.
[17] M. Magdon-Ismail and J. Purnell, "SSDE-cluster: Fast overlapping clustering of networks using sampled spectral distance embedding and GMMs," in *2011 IEEE 3rd International Conference on Privacy, Security, Risk and Trust and 2011 IEEE 3rd International Conference on Social Computing (PASSAT/SocialCom 2011)*, 2011, pp. 756–759.
[18] J. J. Whang, I. S. Dhillon, and D. F. Gleich, "Non-exhaustive, overlapping k-means," in *Proc. of the 2015 SIAM International Conference on Data Mining (SDM 2015)*, 2015, pp. 936–944.
[19] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social Networks*, vol. 5, no. 2, pp. 109 – 137, 1983.
[20] J. M. Kumpula, M. Kivelä, K. Kaski, and J. Saramäki, "Sequential algorithm for fast clique percolation," *Physical Review E*, vol. 78, p. 026109, 2008.
[21] I. Farkas, D. Abel, G. Palla, and T. Vicsek, "Weighted network modules," *New Journal of Physics*, vol. 9, no. 6, p. 180, 2007.
[22] H.-W. Shen, X. Q. Cheng, and J.-F. Guo, "Quantifying and identifying the overlapping community structure in networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2009, no. 07, p. P07042, 2009.
[23] I. Psorakis, S. Roberts, M. Ebden, and B. Sheldon, "Overlapping community detection using bayesian non-negative matrix factorization," *Physical Review E*, vol. 83, p. 066114, 2011.
[24] D. Jin, B. Gabrys, and J. Dang, "Combined node and link partitions method for finding overlapping communities in complex networks," *Scientific reports*, vol. 5, p. 8600, 2015.
[25] W. Chen, Z. Liu, X. Sun, and Y. Wang, "A game-theoretic framework to identify overlapping communities in social networks," *Data Mining and Knowledge Discovery*, vol. 21, no. 2, pp. 224–240, 2010.
[26] J. Xie and B. K. Szymanski, "Towards linear time overlapping community detection in social networks," in *Advances in Knowledge Discovery and Data Mining (PAKDD 2012)*, 2012, pp. 25–36.
[27] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
[28] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *Journal of Research of the National Bureau of Standards*, vol. 45, no. 4, pp. 255–282, 1950.
[29] M. Newman, *Networks: an introduction*. Oxford university press, 2010.
[30] U. Brandes, "A faster algorithm for betweenness centrality," *The Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.
[31] S. White and P. Smyth, "A spectral clustering approach to finding communities in graphs," in *Proc. of the 2005 SIAM International Conference on Data Mining (SDM 2005)*, 2005, pp. 274–285.
[32] T. M. Cover and J. A. Thomas, *Elements of information theory*, 2nd ed. Willey-Interscience, 2006.
[33] A. Condon and R. M. Karp, "Algorithms for graph partitioning on the planted partition model," *Random Structures Algorithms*, vol. 18, no. 2, pp. 116–140, 2001.
[34] I. J. Good, "On the application of symmetric dirichlet distributions and their mixtures to contingency tables," *The Annals of Statistics*, vol. 4, no. 6, pp. 1159–1189, 1976.
[35] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.
[36] G. R. Kiss, C. Armstrong, R. Milroy, and J. Piper, "An associative thesaurus of English and its computer analysis," *The computer and literary studies*, pp. 153–165, 1973.
[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. of the 26th International Conference on Neural Information Processing Systems (NIPS 2013)*, 2013, pp. 3111–3119.
[38] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.
[39] J. Kunegis, "KONECT: The koblenz network collection," in *Proc. of the 22nd International Conference on World Wide Web (WWW 2013)*, 2013, pp. 1343–1350.
[40] J. Leskovec and R. Sosič, "SNAP: A general-purpose network analysis and graph-mining library," *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 1, pp. 1:1–1:20, 2016.

Hadrien Van Lierde (GSM17) received the BSc and M. Eng. degrees in applied mathematics from Universite catholique de Louvain, Louvain-la-Neuve, Belgium, in 2015. He is currently working at the City University of Hong Kong, Hong Kong, toward the PhD degree in electronic engineering. His research interests include spectral graph theory, complex networks, machine learning and data mining.

Tommy W. S. Chow (M93, SM03, F18) received the B.Sc. (Hons.) and Ph.D. degrees from the Department of Electrical and Electronic Engineering, University of Sunderland, Sunderland, U.K., He is currently a Professor with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong. He has authored or co-authored over 200 technical articles related to his research, five book chapters, and one book. His research interests include neural networks, machine learning and pattern recognition.

Guanrong Chen (M89, SM92, F97) received the MSc degree in Computer Science from Sun Yat-sen University, Guangzhou, China in 1981 and the PhD degree in Applied Mathematics from Texas A&M University, College Station, Texas in 1987. He has been a Chair Professor and the Founding Director of the Centre for Chaos and Complex Networks at the City University of Hong Kong since year 2000, prior to that he was a tenured Full Professor at the University of Houston, Texas, USA. He was awarded the 2011 Euler Gold Medal, Russia, and conferred Honorary Doctorate by the Saint Petersburg State University, Russia in 2011 and by the University of Le Havre, Normandie, France in 2014. He is a Member of the Academy of Europe and a Fellow of The World Academy of Sciences, and is a Highly Cited Researcher in Engineering as well as in Mathematics according to Thomson Reuters.