Dynamic Bi-Colored Graph Partitioning

He, Yanbin; Coutino, Mario; Isufi, Elvin; Leus, Geert

Publication date
2022

Document Version
Final published version

Published in
30th European Signal Processing Conference, EUSIPCO 2022 - Proceedings

Dynamic Bi-Colored Graph Partitioning

Yanbin He[†], Mario Coutino[‡], Elvin Isufi[†], Geert Leus[†]
[†]*Delft University of Technology, Delft, The Netherlands*
[‡]*Radar Technology, TNO, Den Haag, The Netherlands*

***Abstract*—In this work, we focus on partitioning dynamic graphs with two types of nodes (bi-colored), though not necessarily bipartite graphs. They commonly appear in communication network applications, e.g., one color being base stations, the other users, and the dynamic process being the varying connection status between base stations and moving users. We introduce a partition cost function that incorporates the coloring of the graph and propose solutions based on the generalized eigenvalue problem (GEVP) for the static two-way partition problem. The static multi-way partition problem is then handled by a heuristic based on the two-way partition problem. Regarding the adaptive partition, an eigenvector update-based method is proposed. Numerical experiments demonstrate the performance of the devised approaches.**

***Index Terms*—graph partitioning, spectral clustering, generalized eigenvalue problem, dynamic graphs**

I. INTRODUCTION

Graphs have been gaining more and more attention since they can model networked data and their complex relations [23]. They are pervasive and present in several real-world applications, e.g., social media [16], the internet [4], and sensor networks [7]. The graph structure can be used to cluster (or partition) a network in subgroups that exhibit a certain type of *closeness*. For example, by performing clustering on graphs, we can identify groups of nodes sharing certain similarities [24], which has benefits for certain applications. In long-term evolution cellular networks, for instance, the capacity can be improved by performing clustering based on network connectivity [6]. Another example can be found in a computer with multiple processors, where an appropriate clustering of the processors can result in a good load balancing and minimize the ratio of communication over computation cost for a given task [17].

In the literature, various clustering techniques have been proposed. Among them, spectral clustering has become one of the most popular clustering algorithms [22]. Spectral clustering utilizes the information carried by the eigenvectors of the graph Laplacian to partition the vertices of the graph [10]. Spectral clustering is conceptually simple and can be solved efficiently by linear algebra methods [21] and often achieves better results compared with traditional methods [22].

To derive appropriate and meaningful clusters, a well-chosen criterion is required, which can be translated into a cost function that needs to be optimized. Some well-known cost functions, such as *MinMaxCut* [5], [12] and *NormalizedCut* [19], are generally applicable and capture metrics of interest for partitioning typical graphs. However, for some applications, we might be interested in specific goals rendering general cost functions not applicable. For example, in a communication network where there exist different types of nodes (colors), e.g., users and base stations, typical cost functions for graph partitioning are oblivious to the graph *coloring*. Furthermore, for some applications, graphs as models should be evolving in time, i.e., they should be dynamic [13]. These dynamics can be variations of edge weights, or the number of nodes within the graph. As partitioning dynamic graphs from scratch might incur a high computational burden, adaptively updating graph clusters becomes of utmost importance.

At the time of performing this work, Yanbin was partially supported by the ME-MSc scholarship from TU Delft and Mario Coutino by CONACYT. The work of E. Isufi is supported by the TU Delft AI Labs programme. emails: {y.he-1, e.isufi-1, g.j.t.leus}@tudelft.nl, mario.coutinominguez@tno.nl.

In our work, we focus on partitioning dynamic bi-colored graphs, where there are two types of vertices and the dynamics are related to variations of edge weights. Taking into account the different node types, we put forth a cost function capturing the color dependency of the partition which is relevant to typical communication network clustering problems (Sec. II). To tackle the partition problem, we first focus on the static two-way partitioning problem (bisection), which is a special case of the multi-way partition problem, and provide a way to relax the introduced cost function relying on spectral clustering. This approach is then extended to the multi-way partition problem (Sec. III). Finally, based on the discussions on the static case, an adaptive partitioning method is introduced, using the eigenvectors update approach based on matrix perturbation theory (Sec. IV).

II. BI-COLORED GRAPH PARTITIONING

Let us consider a (possibly) dynamic graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with adjacency matrix $\mathbf{W}$ and graph Laplacian $\mathbf{L}$, where $\mathcal{V}$ and $\mathcal{E}$ are the vertex set and edge set, respectively. We denote $|\mathcal{V}| = N$ and assume there are *two* types of vertices (colors) $\mathcal{B}$ and $\mathcal{U}$ in the graph, satisfying $\mathcal{B} \cup \mathcal{U} = \mathcal{V}$ and $\mathcal{B} \cap \mathcal{U} = \varnothing$. An edge $e_{i,j} \in \mathcal{E}$ can exist between the *same* and *different* types of vertices and its edge weight is denoted as $[\mathbf{W}]_{i,j} = w_{i,j} \geq 0$. Thus, this graph is *not* necessarily bipartite.

Now, let us consider a 4G/5G cellular network to motivate the (dynamic) bi-colored graph partition problem. When the coordinated multipoint concept is used to improve the performance, data has to be exchanged between base stations [2], [3], [9], [11], [26]. This data, e.g., channel state information or user data, is shared through the backhaul network. However, exchanges through the whole network lead to a signaling overhead and are thus infeasible. Therefore, the data should be exchanged locally [1], [25], that is, we need to identify local clusters and communicate the information within each cluster. In this case, we have two types of nodes (colors) given by base stations and users and two types of edges with varying weights, including wireless links (base stations - users) and wired links (base stations - base station). As the users are mobile, the connection status between users and base station changes over time (dynamic).

In such a setting, it is desired to minimize the communication across local clusters, since the traffic routed between different clusters is expensive [8], [14]. In addition, to use base stations efficiently, clusters need to be as dense as possible regarding wireless links (base stations - users). These goals can be formally written as follows: given a number of clusters K, our objective is to decompose the vertices of the dynamic graph into K disjoint and non-empty clusters $\{\mathcal{C}_m\}_{m=1,\ldots,K}$ by minimizing the following cost function

$$J(\{\mathcal{C}_m\}) = \sum_{m=1}^{K} \frac{\sum_{i\in\mathcal{C}_m, j\in\mathcal{V}\setminus\mathcal{C}_m} w_{i,j}}{\sum_{i\in\mathcal{B}_m, j\in\mathcal{U}_m} w_{i,j}}, \tag{1}$$

where $\mathcal{B}_m = \mathcal{C}_m \cap \mathcal{B}$ $(\neq \varnothing)$, $\mathcal{U}_m = \mathcal{C}_m \cap \mathcal{U}$ $(\neq \varnothing)$ are non-empty set of nodes of the same color in each cluster $\mathcal{C}_m$ and $\mathcal{C}_i \cap \mathcal{C}_j = \varnothing$ implies there are no edges between two clusters, $\forall i, j \in \{1, ..., K\}, i \neq j$. In other words, we want to minimize the connections between the different clusters (captured in the numerator) yet maximize the connections between the different types of nodes within every cluster (captured in the denominator). For the dynamic setting, this problem needs to be solved as the graph evolves.

III. Static Two-Way Partitioning Problem

To approach the adaptive multi-way clustering problem, we first focus on the simpler *static graph bisection problem*, which is a special case with $K = 2$. As for classical graph partitioning, graph bisection sheds light on how to deal with the multi-way partition problem. After the static case, we will go to the dynamic case.

A. Matrix-Vector Form of the Cost Function

To rearrange cost (1) in a more manageable form, let us define the cluster indicator vector for set $\mathcal{C}_m$ as $\mathbf{c}_m \in \{0,1\}^N$, i.e., $[\mathbf{c}_m]_i = 1$ if $i \in \mathcal{C}_m$. Further, since $\mathbf{c}_m \neq \mathbf{0}$, $\forall m$ and $\sum_{m=1}^{K} \mathbf{c}_m = \mathbf{1}$, we could write the numerator of (1) as

$$\sum_{i\in\mathcal{C}_m, j\in\mathcal{V}\setminus\mathcal{C}_m} w_{i,j} = \mathbf{c}_m^\top \mathbf{W}(\mathbf{1} - \mathbf{c}_m) \overset{\text{(a)}}{=} \mathbf{c}_m^\top \mathbf{L}\mathbf{c}_m,$$

where (a) is due to $\mathbf{c}_m^\top \mathbf{W}\mathbf{1} = \mathbf{c}_m^\top \text{diag}(\mathbf{D}) = \mathbf{c}_m^\top \mathbf{D}\mathbf{c}_m$ since $\mathbf{c}_m$ is a binary vector. Likewise, let us define the two color indicator vectors $[\mathbf{b}]_i = 1$ if $i \in \mathcal{B}$ and $[\mathbf{u}]_i = 1$ if $i \in \mathcal{U}$ and the respective indicator matrices $\mathbf{B}_{\text{in}} := \text{diag}(\mathbf{b}) \in \{0,1\}^{N\times N}$ and $\mathbf{U}_{\text{in}} := \text{diag}(\mathbf{u}) \in \{0,1\}^{N\times N}$ to write the denominator of (1) as

$$\sum_{i\in\mathcal{B}_m, j\in\mathcal{U}_m} w_{i,j} = \mathbf{c}_m^\top \mathbf{B}_{\text{in}} \mathbf{W} \mathbf{U}_{\text{in}} \mathbf{c}_m = \mathbf{c}_m^\top \tilde{\mathbf{W}} \mathbf{c}_m.$$

where we defined $\tilde{\mathbf{W}} := \mathbf{B}_{\text{in}}\mathbf{W}\mathbf{U}_{\text{in}}$ for a compact notation. Bringing these together, the cost function (1) becomes

$$J(\{\mathbf{c}_m\}) = \sum_{m=1}^{K} \frac{\mathbf{c}_m^\top \mathbf{L}\mathbf{c}_m}{\mathbf{c}_m^\top \tilde{\mathbf{W}}\mathbf{c}_m}. \tag{2}$$

For $K = 2$, one indicator vector is the direct complement of the other, i.e., $\mathbf{c}_2 = \mathbf{1} - \mathbf{c}_1$. Hence, we can use a single variable denoted as $\mathbf{c}$. The cost then becomes

$$J(\mathbf{c}) = \frac{\mathbf{c}^\top \mathbf{L}\mathbf{c}}{\mathbf{c}^\top \tilde{\mathbf{W}}\mathbf{c}} + \frac{\mathbf{c}^\top \mathbf{L}\mathbf{c}}{\kappa - \mathbf{q}^\top \mathbf{c} + \mathbf{c}^\top \tilde{\mathbf{W}}\mathbf{c}}, \tag{3}$$

where $\kappa := \mathbf{1}^\top \tilde{\mathbf{W}}\mathbf{1}$, $\mathbf{q} := (\tilde{\mathbf{W}}^\top + \tilde{\mathbf{W}})\mathbf{1}$, and $\mathbf{c} \in \{0,1\}^N$ with $\mathbf{c} \neq \mathbf{0}, \mathbf{1}$.

B. Solution based on the GEVP

Similar to previous works [5], [22], to simplify the optimization, we drop the binary constraints on $\mathbf{c}$ and relax it to $\mathbb{R}^N$, which is then called *the continuous indicator vector*. Clearly, this cost function is *scale sensitive* in $\mathbf{c}$ because of the existence of the linear term. Furthermore, the solutions $\mathbf{c} = \mathbf{0}$ or $\mathbf{1}$ should be avoided. To deal with these issues, we adopt the constraint $\mathbf{c}^\top \mathbf{L}\mathbf{c} = 1$ on $\mathbf{c}$. In summary, we focus on the following optimization problem:

$$\min_{\mathbf{c}\in\mathbb{R}^N} J(\mathbf{c}) \quad \text{s.t.} \quad \mathbf{c}^\top \mathbf{L}\mathbf{c} = 1. \tag{4}$$

To partition the graph, we first provide the critical points of (4).

Proposition 1. Consider the optimization problem (4). Its critical points correspond to vectors $\mathbf{c}$ satisfying the expression

$$\mathbf{W}_{\text{S}}(\mathbf{c} - 0.5\gamma\mathbf{1}) = \lambda\mathbf{L}(\mathbf{c} - 0.5\gamma\mathbf{1}), \tag{5}$$

where

$$\lambda = \frac{4b_1 b_2(b_1 + b_2 + \nu b_1 b_2)}{a(b_1^2 + b_2^2)}, \quad \gamma = \frac{2b_1^2}{b_1^2 + b_2^2},$$

with $a := \mathbf{c}^\top \mathbf{L}\mathbf{c}$, $b_1 := \mathbf{c}^\top \tilde{\mathbf{W}}\mathbf{c}$, $b_2 := \kappa - \mathbf{q}^\top \mathbf{c} + \mathbf{c}^\top \tilde{\mathbf{W}}\mathbf{c}$, $\nu = -\kappa(b_1^2 + b_2^2)^{-1}$ the optimal Lagrange multiplier, and $\mathbf{W}_{\text{S}} := 2(\tilde{\mathbf{W}}^\top + \tilde{\mathbf{W}})$.

Proof. Let us start from the Lagrangian $\mathcal{L}(\mathbf{c}, \nu) = J(\mathbf{c}) + \nu(\mathbf{c}^\top \mathbf{L}\mathbf{c} - 1)$ with Lagrangian multiplier $\nu \in \mathbb{R}$. Taking the gradient of this Lagrangian w.r.t. $\mathbf{c}$, and equating it to zero, we obtain the expression

$$\mathbf{W}_{\text{S}}\mathbf{c} = \lambda\mathbf{L}\mathbf{c} + \gamma\mathbf{q}, \tag{6}$$

where λ and γ are defined as in the proposition based on the earlier definitions of a, b_1, and b_2. Note that at this point, we did not yet determine the Lagrangian parameter ν which will be discussed later. Taking the structure of $\mathbf{q} = (\tilde{\mathbf{W}}^\top + \tilde{\mathbf{W}})\mathbf{1}$ into consideration, we can further simplify (6) to

$$\mathbf{W}_{\text{S}}(\mathbf{c} - 0.5\gamma\mathbf{1}) = \lambda\mathbf{L}\mathbf{c}.$$

Since $\mathbf{1}$ is in the kernel of $\mathbf{L}$, we then also have

$$\mathbf{W}_{\text{S}}(\mathbf{c} - 0.5\gamma\mathbf{1}) = \lambda\mathbf{L}(\mathbf{c} - 0.5\gamma\mathbf{1}), \tag{7}$$

which is a generalized eigenvalue problem (GEVP). And if we denote the generalized eigenvector of (7) as $\mathbf{u}$ satisfying $\mathbf{u}^\top \mathbf{W}_\mathrm{S}\mathbf{u} = \lambda, \mathbf{u}^\top \mathbf{L}\mathbf{u} = 1$, then we obtain $\mathbf{c} = \mathbf{u} + 0.5\gamma\mathbf{1}$.

To show that the vectors $\mathbf{c}$ derived by the generalized eigenvectors $\mathbf{u}$ are valid critical points of (3), we have to demonstrate that (i) $\mathbf{c}$ should satisfy $\mathbf{c}^\top \mathbf{L}\mathbf{c} = 1$, which is guaranteed by $\mathbf{u}^\top \mathbf{L}\mathbf{u} = 1$ as $\mathbf{1}$ is in the kernel of $\mathbf{L}$, and (ii) $\mathbf{c}$ should lead to an equality between λ defined in (5) and the generalized eigenvalue. This can be ensured by having $\nu = -\kappa(b_1^2 + b_2^2)^{-1}$. If we equate the way to compute the generalized eigenvalue (left) and λ (right)

$$\frac{\mathbf{u}^\top \mathbf{W}_\mathrm{S}\mathbf{u}}{\mathbf{u}^\top \mathbf{L}\mathbf{u}} = \frac{4b_1^2 b_2 + 4b_1 b_2^2 + 4\nu b_1^2 b_2^2}{a(b_1^2 + b_2^2)}, \tag{8}$$

and plug in $\mathbf{c} = \mathbf{u} + 0.5\gamma\mathbf{1}$, then by some manipulations, we obtain $\nu = -\kappa(b_1^2 + b_2^2)^{-1}$. The resulting vectors $\mathbf{c}$ are thus the critical points of (4). □

To retrieve the continuous indicator vector $\mathbf{c}$, we first have to solve the GEVP (7). However, as the Laplacian, by its mere definition, has generally rank $N-1$, this GEVP has an infinite eigenvalue [20]. Since we only care about the finite ones, we can simply reshape the size of the singular value matrix of $\mathbf{L}$, $\mathbf{\Sigma_L}$, to $(N-1) \times (N-1)$ by eliminating the zero eigenvalue and then constructing $\tilde{\mathbf{A}} = (\mathbf{U_L}\mathbf{\Sigma_L}^{-1/2})^\top \mathbf{W}_\mathrm{S}(\mathbf{U_L}\mathbf{\Sigma_L}^{-1/2})$. Here, $\mathbf{U_L}$ collects the appropriate $N-1$ eigenvectors of $\mathbf{L}$. The generalized eigenvectors are then constructed as

$$\mathbf{U} = \mathbf{U_L}\mathbf{\Sigma_L}^{-1/2}\mathbf{U}_{\tilde{\mathbf{A}}}, \tag{9}$$

where $\mathbf{U}_{\tilde{\mathbf{A}}}$ denotes the eigenvector matrix of $\tilde{\mathbf{A}}$. Note that now we obtain a matrix $\mathbf{U}$ with size $N \times (N-1)$ and it can be easily seen that $\mathbf{U}$ satisfies

$$\mathbf{U}^\top \mathbf{W}_\mathrm{S}\mathbf{U} = \mathbf{\Lambda},\ \mathbf{U}^\top \mathbf{L}\mathbf{U} = \mathbf{I},$$

where $\mathbf{\Lambda} \in \mathbb{R}^{(N-1)\times(N-1)}$ collects the finite generalized eigenvalues on its diagonal.

Although to compute $\mathbf{c}$ we need to know γ (which depends on $\mathbf{c}$), we show that this does not pose a problem for finding the partitions. But as we have discussed, the cost function has multiple critical points related to multiple eigenvectors. To determine which one to pick, we introduce the next result.

Proposition 2. The cost function can be upper-bounded by

$$J(\mathbf{c}) < \frac{8}{\lambda}. \tag{10}$$

Proof. We have

$$\lambda = \frac{4b_1 b_2(b_1 + b_2 + \nu b_1 b_2)}{a(b_1^2 + b_2^2)} = \frac{4(b_1 + b_2)^2}{J(\mathbf{c})(b_1^2 + b_2^2)} + \frac{4\nu_1^2 b_2^2}{b_1^2 + b_2^2}.$$

With the Cauchy–Schwarz inequality, we can obtain

$$\lambda \leq \frac{8}{J(\mathbf{c})} - 4\kappa\Big(\frac{b_1 b_2}{b_1^2 + b_2^2}\Big)^2 < \frac{8}{J(\mathbf{c})},$$

since the second term is a positive value. By exchanging the position of λ and $J(\mathbf{c})$, the upper bound (10) can be obtained. □

Therefore, the generalized eigenvector corresponding to the largest eigenvalue should be chosen as the continuous indicator vector as it tends to produce the smallest cost function by minimizing the upper bound on $J(\mathbf{c})$. After this selection, similarly to classical graph partitioning, a binary solution, $\mathbf{c}^* \in \{0,1\}^N$, can be retrieved by thresholding the continuous solution through a function value sweep finding the best function evaluation. Since $0.5\gamma\mathbf{1}$ is only an offset, working on $\mathbf{u}$ produces the same clusters as working on $\mathbf{c}$. Thus, there is no need to compute γ. From now on, we will not differentiate $\mathbf{u}$ and $\mathbf{c}$.

C. Static Multi-way Partition

Similar to classical graph partitioning, for the multi-way partitioning problem, we follow a heuristic approach and select the K generalized eigenvectors related to the K largest eigenvalues. One can stack these K vectors into a matrix $\mathbf{U}_K$, then treat each row of $\mathbf{U}_K$ as coordinates and finally apply K-means to obtain a label for each node.

IV. Adaptive Partitioning

We proceed to derive the adaptive partitioning for when the graph is dynamic. We formulate the variations in the graph topology as perturbations to the graph matrices. The main idea is to use the current eigenpair to approximate the new ones for the perturbed graph and update accordingly the clusters. Thus, although the method starts from the exact eigenpairs, during the update procedure, there is no need to compute the eigenpairs from scratch, reducing the computational load.

Assume the m-th eigenvector $\mathbf{u}_m$ and its related eigenvalue λ_m are known and they are derived from the following GEVP

$$\mathbf{W}_\mathrm{S}\mathbf{u}_m = \lambda_m \mathbf{L}\mathbf{u}_m. \tag{11}$$

Further consider the perturbed eigensystem [13]

$$\begin{aligned}&(\mathbf{W}_\mathrm{S} + \Delta\mathbf{W}_\mathrm{S})(\mathbf{u}_m + \Delta\mathbf{u}_m)\\ =&(\lambda_m + \Delta\lambda_m)(\mathbf{L} + \Delta\mathbf{L})(\mathbf{u}_m + \Delta\mathbf{u}_m).\end{aligned} \tag{12}$$

Now, to update the m-th eigenpair we need to derive $\Delta\mathbf{u}_m$ and $\Delta\lambda_m$ given $\Delta\mathbf{W}_\mathrm{S}$ and $\Delta\mathbf{L}$. $\Delta\lambda_m$ can be given by

$$\Delta\lambda_m = \frac{\mathbf{u}_m^\top(\Delta\mathbf{W}_\mathrm{S} - \lambda_m\Delta\mathbf{L})(\mathbf{u}_m + \Delta\mathbf{u}_m)}{\mathbf{u}_m^\top(\mathbf{L} + \Delta\mathbf{L})(\mathbf{u}_m + \Delta\mathbf{u}_m)} \tag{13}$$

through expanding (12). To find $\Delta\mathbf{u}_m$, we need to solve the linear system resulting from expanding (12), i.e.,

$$\begin{aligned}&(\mathbf{W}_\mathrm{S} + \Delta\mathbf{W}_\mathrm{S} - (\lambda_m + \Delta\lambda_m)(\mathbf{L} + \Delta\mathbf{L}))\Delta\mathbf{u}_m\\ =&(\Delta\lambda_m\mathbf{L} + \lambda_m\Delta\mathbf{L} + \Delta\lambda_m\Delta\mathbf{L} - \Delta\mathbf{W}_\mathrm{S})\mathbf{u}_m\end{aligned} \tag{14}$$

which can be written as $\tilde{\mathbf{K}}\Delta\mathbf{u}_m = \tilde{\mathbf{h}}$.

Since $\tilde{\mathbf{K}}$ is a singular matrix, which can be seen by rearranging (12) as $\tilde{\mathbf{K}}(\mathbf{u}_m + \Delta\mathbf{u}_m) = 0$, a unique $\Delta\mathbf{u}_m$ cannot be determined. We assume here that the perturbations of the matrices are small enough such that the approximation of the generalized eigenvectors and eigenvalues hold. This provides us with the intuition that the $\Delta\mathbf{u}_m$ must be small in the l_2-norm sense as well. Therefore, we can add the norm of the perturbation as a regularization term when solving the related

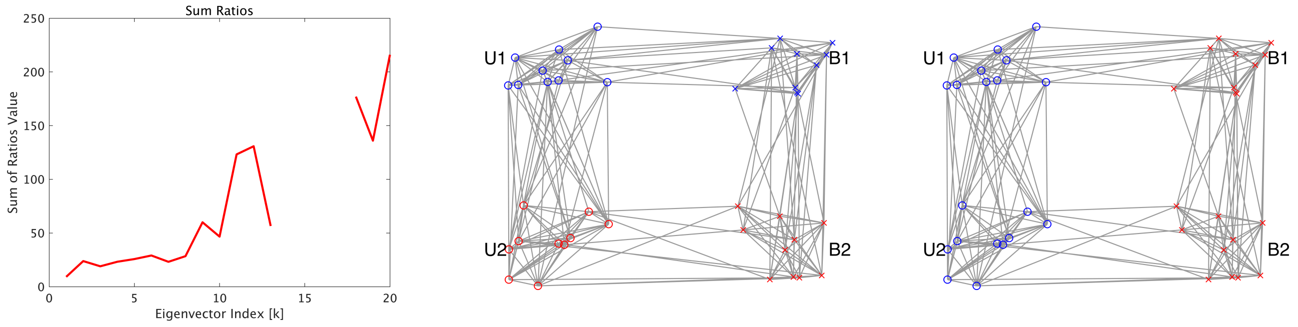

Fig. 1: Red and blue: clusters. U1, U2, B1, B2: 4 blocks; U (circle), B (cross): two types of nodes. Left: Cost function value for generalized eigenvectors (eigenvalues from large to small). Middle: Bisection of the proposed method. Right: Bisection of spectral clustering.

least squares problem. As a result, $\Delta\mathbf{u}_m$ can be retrieved by solving the following optimization problem:

$$\min_{\Delta\mathbf{u}_m} ||\tilde{\mathbf{K}}\Delta\mathbf{u}_m - \tilde{\mathbf{h}}||_2^2 + \mu||\Delta\mathbf{u}_m||_2^2, \tag{15}$$

where $\mu > 0$ is a regularization parameter. This is the well-known Tikhonov regularization problem, which can be solved efficiently using conjugate gradient descent (CGD) with i_{max} iterations [18].

So far, we explained the different steps of the eigenpair updating method. To perform K-way partitioning, the first K eigenvectors are required. We initialize the method with the eigenpairs computed by the GEVP. When the graph changes, we first set $\Delta\mathbf{u}_m = \mathbf{0}$ and use (13) to obtain the first approximation of $\Delta\lambda_m$ with $\mathcal{O}(|\mathcal{E}|)$ operations and then we solve (15) based on $\Delta\lambda_m$ to obtain the first approximation of $\Delta\mathbf{u}_m$ with $\mathcal{O}(|\mathcal{E}|)$ operations as well. We perform these two steps alternatively with I iterations to refine the approximations for each eigenpair. However, this method is incapable of ensuring that the tracked K eigenvectors are the ones related to the K largest eigenvalues. To address this issue, one can choose to track $M \geq K$ eigenvectors. And after each update, one reorders the eigenvalues and the eigenvectors descendingly as the first K largest eigenvectors are sufficient and required to produce the desired clusters. When $\mathbf{L}$ is a sparse matrix, the overall complexity of the update is $\mathcal{O}(M \times I \times i_{max} \times |\mathcal{E}|)$.

V. Numerical Results

A. Static Case

We consider a graph generated using a *stochastic block model* with 4 blocks, *U1*, *U2*, *B1*, and *B2*, to be bisected [15]. Each block has 10 nodes. The intra-block edge probability is 0.9. The edge probability between U1-U2 and B1-B2 is 0.2. The edge probability between U1-B1 and U2-B2 is 0.1. The goal is to cluster U1-B1 and U2-B2.

We consider the binary adjacency matrix as $\mathbf{W}$, and $\mathbf{L}$ as its combinatorial Laplacian. The (continuous) indicator vectors are derived by the procedure discussed in Section III-B. In the implementation, we construct $\mathbf{u}$ by computing the eigenvectors of $\mathbf{L}$ and $\tilde{\mathbf{A}}$ related to eigenvalues larger than 10^{-6}. The cost function values obtained by the *thresholded* columns of $\mathbf{U}$ are shown in Fig. 1 (left), where the gap is because the cost is infinity when being evaluated by some of the discretized columns. The bisection result is shown in Fig. 1 (middle). As a comparison, spectral clustering (SC) is also performed as in [22] which arrives at different clusters as illustrated in Fig. 1 (right).

When using spectral clustering, we intrinsically minimize a cost function such as *MinMaxCut* with $\sum_{i\in\mathcal{C}_m, j\in\mathcal{C}_m} w_{i,j}$ in the denominator, i.e., the cluster density is color-independent. But our proposed cost function only considers the edges between a U block and a B block. Consequently, the dense intra-block edges and edges between U1-U2/B1-B2 have no effect when producing clusters, leading to the desirable result. To summarize, the results in this section validate our choice of eigenvectors, and the solution to generate clusters.

B. Adaptive Case

The graph considered here is the model of a simulated cellular network. The cellular network has 37 base stations and 3 hotspots with a Gaussian shape. Each hotspot has 400 users. The base stations are initially uniformly distributed with inter site distance (ISD) 1Km. The heterogeneity in the layout of the base stations is achieved by randomly moving each base station to another position within a 400 m $\times$ 400 m square centered around its original position. The layout of the cellular network is shown in Fig. 2 (Left). We assign graph weights as $w_{i,j} = 1/d_{i,j}$, where $d_{i,j}$ is the Euclidean distance between nodes i and j.

As connection is assumed to be local, we only preserve edges having weights larger than a threshold ϵ. For different types of edges, we set two thresholds.

- For the edges between different base stations[1]:

$$\epsilon = 1/(1.1 \times \text{ISD}).$$

- For the edges between users and base stations:

$$\epsilon = 1/(1.1 \times r_c),$$

with $r_c = \text{ISD}/\sqrt{3}$ the cell radius.

In the simulations, the regularization parameter μ is 10^{-4}, the number of alternating steps I is 2, the maximum number of iterations in CGD i_{max} is $6\sqrt{N}$, and the number of tracked components M is 4. The graph dynamics result from randomly changing each user's position within a 200 m $\times$ 200

[1]The factor 1.1 is used to account for the network heterogeneity.

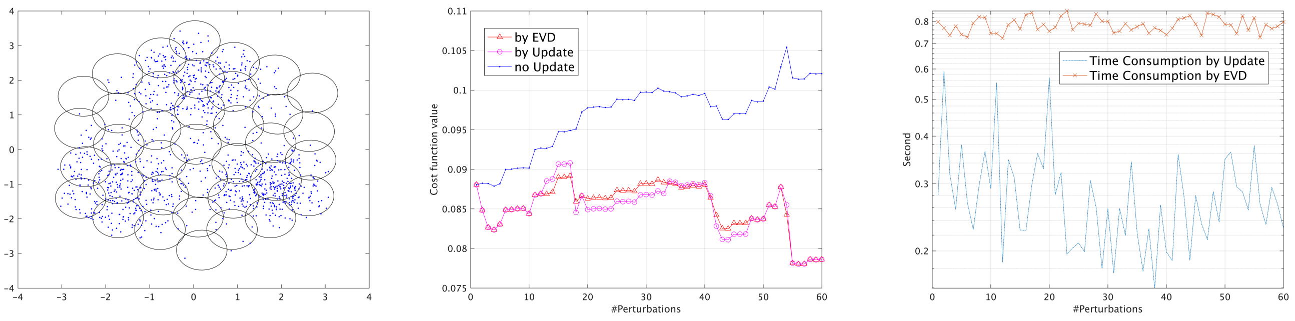

Fig. 2: Left: Cellular network layout. Middle: Cost function. Right: Time consumption of each update. [by EVD]: using (9); [by Update]: using the proposed method; [no Update]: using the initial clusters.

m square centered around its previous position. Within each perturbation, 20 users change their positions. Given that there are 1200 users in the network, the change in the graph caused by moving users can be modeled as a series of perturbations of the adjacency matrix $\{\Delta \mathbf{W}_t\}_{t=1,2,\ldots,60}$. We update the eigenvectors and the clusters after each perturbation based on the estimates after the previous perturbation (initialized by the exact ones). We illustrate the cost function value and the computation time at each time instant in Fig. 2 (middle) and in Fig. 2 (right), respectively, where we can see how the update procedure performs when the network changes over time. As we can observe, the cost function from the proposed method (Update) is almost the same as solving (11) exactly (EVD), indicating a similarity in the determined clusters, but with much less processing time.

VI. Conclusions

We studied the partitioning problem for dynamic bi-colored graphs which contain two types of vertices. In the proposed cost function, the color dependency was considered. We first detailed the relaxation of the bisection problem and propose a solution based on the generalized eigenvalue decomposition. This method is subsequently extended to a multi-way partitioning approach. After that, we introduced an adaptive partitioning method for dynamic graphs based on matrix perturbation theory and incremental clustering principles. Finally, we validated the devised methods over synthetic bi-colored graphs.

References

[1] S. Bassoy, H. Farooq, M. A. Imran, and A. Imran, “Coordinated multipoint clustering schemes: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 743–764, 2017.

[2] S. Bassoy, M. Jaber, M. A. Imran, and P. Xiao, “Load aware self-organising user-centric dynamic comp clustering for 5g networks,” *IEEE Access*, vol. 4, pp. 2895–2906, 2016.

[3] T. Biermann, L. Scalia, C. Choi, H. Karl, and W. Kellerer, “Comp clustering and backhaul limitations in cooperative cellular mobile access networks,” *Pervasive and Mobile Computing*, vol. 8, no. 5, pp. 662–681, 2012.

[4] P. Desikan, N. Pathak, J. Srivastava, and V. Kumar, “Incremental page rank computation on evolving graphs,” in *Special interest tracks and posters of the 14th International Conference on World Wide Web*, 2005, pp. 1094–1095.

[5] M. Gu, H. Zha, C. Ding, X. He, H. Simon, and J. Xia, “Spectral relaxation models and structure analysis for k-way graph clustering and bi-clustering,” 2001.

[6] M. Hajjar, G. Aldabbagh, and N. Dimitriou, “Using clustering techniques to improve capacity of lte networks,” in *2015 21st Asia-Pacific Conference on Communications (APCC)*. IEEE, 2015, pp. 68–73.

[7] I. Jabłoński, “Graph signal processing in applications to sensor networks, smart grids, and smart cities,” *IEEE Sensors Journal*, vol. 17, no. 23, pp. 7659–7666, 2017.

[8] A. Lisser and F. Rendl, “Graph partitioning using linear and semidefinite programming,” *Mathematical Programming*, vol. 95, no. 1, pp. 91–101, 2003.

[9] J.-M. Moon and D.-H. Cho, “Inter-cluster interference management based on cell-clustering in network mimo systems,” in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*. IEEE, 2011, pp. 1–6.

[10] A. Ng, M. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” *Advances in neural information processing systems*, vol. 14, pp. 849–856, 2001.

[11] C. T. Ng and H. Huang, “Linear precoding in cooperative mimo cellular networks with limited coordination clusters,” *IEEE Journal on Selected Areas in communications*, vol. 28, no. 9, pp. 1446–1454, 2010.

[12] F. Nie, C. Ding, D. Luo, and H. Huang, “Improved minmax cut graph clustering with nonnegative relaxation,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2010, pp. 451–466.

[13] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. S. Huang, “Incremental spectral clustering by efficiently updating the eigen-system,” *Pattern Recognition*, vol. 43, no. 1, pp. 113–127, 2010.

[14] K. Park, K. Lee, S. Park, and H. Lee, “Telecommunication node clustering with node compatibility and network survivability requirements,” *Management Science*, vol. 46, no. 3, pp. 363–374, 2000.

[15] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, “GSPBOX: A toolbox for signal processing on graphs,” *ArXiv e-prints*, Aug. 2014.

[16] M. A. Russell, *Mining the social web: Analyzing data from Facebook, Twitter, LinkedIn, and other social media sites.* ” O’Reilly Media, Inc.”, 2011.

[17] Y. Saad, *Iterative methods for sparse linear systems*. SIAM, 2003.

[18] J. R. Shewchuk *et al.*, “An introduction to the conjugate gradient method without the agonizing pain,” 1994.

[19] 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.

[20] G. W. Stewart, “Matrix perturbation theory,” 1990.

[21] N. Tremblay and A. Loukas, “Approximating spectral clustering via sampling: a review,” *Sampling Techniques for Supervised or Unsupervised Tasks*, pp. 129–183, 2020.

[22] U. Von Luxburg, “A tutorial on spectral clustering,” *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[23] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, “A comprehensive survey on graph neural networks,” *IEEE transactions on neural networks and learning systems*, 2020.

[24] D. Xu and Y. Tian, “A comprehensive survey of clustering algorithms,” *Annals of Data Science*, vol. 2, no. 2, pp. 165–193, 2015.

[25] J. Zhao and Z. Lei, “Clustering methods for base station cooperation,” in *2012 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2012, pp. 946–951.

[26] J. Zhao, T. Q. Quek, and Z. Lei, “Coordinated multipoint transmission with limited backhaul data transfer,” *IEEE Transactions on Wireless Communications*, vol. 12, no. 6, pp. 2762–2775, 2013.