Distributed Link Removal Using Local Estimation of Network Topology

Azwirman Gusrialdi, Member, IEEE, Zhihua Qu, Fellow, IEEE, and Sandra Hirche, Senior Member, IEEE

Abstract—This paper considers the problem of network structure manipulation in the absence of information on the global network topology. In particular, the problem of removing some of links is investigated in order to slow or stop the spread of disease in a network while preserving its connectivity. Existing methods solve this combinatorial problem in a centralized manner and they require the global information of network structure. In this paper, we propose a distributed design algorithm to compute a suboptimal solution to this problem efficiently by mimicking gradient based method, namely by iteratively removing one or multiple links at a time from the network. Specifically, using matrix perturbation analysis we formulate an optimization problem involving the eigenvector associated with the largest eigenvalue of the adjacency matrix and whose solution is equal to a suboptimal solution to the original problem. This strategy also enables us to overcome the combinatorial issue of the problem. Distributed algorithms to estimate the eigenvector and to verify network’s connectivity are then proposed which facilitate us to solve the new optimization problem. In addition, topological insights into the proposed algorithm and optimality of its solution are also discussed. Finally, the proposed distributed design method is demonstrated and evaluated via several numerical examples.

Index Terms—Link removal, largest eigenvalue minimization, distributed algorithm, matrix perturbation.

1 INTRODUCTION

Complex networks as an interaction between individual entities can be found in many real world examples, not only in critical infrastructures such as telecommunication networks, power grids, transportation networks, water distribution networks but also in social networks and biological networks. One important feature in complex networks is its topology which captures the interaction between subsystems/components in a network via sensing, control, communication or physical interconnection. The research on network topology manipulation, including link addition/removal/rewiring, has attracted much interest in recent years, see e.g., [1], [2], [3], [4], [5], [6]. In this paper, we consider the problem of removing a fraction of links from a network such that the largest eigenvalue of the adjacency matrix is minimized while guaranteeing the network connectivity.

As a motivating application, it is known that the disease infecting node in a network can spread through the whole network and may cause outbreak whose speed is related to the network topology and characterized by the largest eigenvalue of its adjacency matrix [7]. Note that the term disease is quite general and is applicable to various applications such that the spread of viruses in networked computer system, cascading failures in networked power grid [8] or attacks in cyber-physical systems. One way to slow the spread of disease or contain its dissemination, other than by immunizing certain individual nodes [9], is by removing a fraction of existing links from the network until the largest eigenvalue of its adjacency matrix is below a certain threshold. This means that we do not need to immunize the individual nodes but rather control their interactions and the disease’s path of spreading [3], [10]. Note that removing links in a network are equivalent to “unfriending” people in social network such as Facebook or disconnecting the physical lines via switches in power network.

In general the problem of link removals of complex network to minimize the largest eigenvalue of the adjacency matrix is NP-hard [1]. Therefore, most of the work focuses on heuristics and approximations of this problem. Although the authors in [3] propose a relaxation method based on semi-definite programming (SDP) to solve this problem, the approach does not scale to large-size networks and is not applicable for weighted networks. The works [1], [10] propose various heuristics such as the one based on eigenvalue sensitivity and number of hop walks of the graph to solve approximately the original problem. However, the authors do not consider connectivity preservation of the network. It is worth noting that minimization of the largest eigenvalue of a symmetric matrix has received a lot of attention since the seventies and several algorithms have been proposed, e.g. [11], [12]. Nonetheless, in this existing work the decision variable is continuous while, in this paper, the decision variable is discrete. Note that extending such a result from continuous variables to discrete ones is far from trivial, because mixed integer optimization problems are often too challenging and could only be solved in special cases. For example, even though it may be possible to apply a relaxation method in combination with a threshold operation to decide which links to be removed, how to optimally choose the threshold is not a trivial problem, especially when taking into account the constraint on the network connectivity. More importantly, all the work mentioned above assumes that the global network structure is known and available to the designer. However, in practice the global network structure information may not be available or may be difficult to accumulate in a centralized fashion because of geographical constraints such as in power grid [13]. Another reason is due to privacy concerns [14], [15], namely each individual node may not be willing to share its set of neighbors to the individuals who are not his/her friend as motivated from social network. This limitation
In this paper, we consider link removal problem for an undirected network in the absence of global network structure in order to control the spread of disease while guaranteeing the network connectivity. The problem is motivated by numerous applications such as: (i) diminishing the propagation of frequency oscillation or cascading failure in power grid while guaranteeing that each load can still be served by the generators; (ii) reducing distribution of malware in social network while ensuring important information can still be received by each person; (iii) removing virus from a computer network while guaranteeing e-mail or message can still be received by each person; (iv) removing virus from a computer network while guaranteeing e-mail or message can still be received by each person; (v) removing virus from a computer network while guaranteeing e-mail or message can still be received by each person; (vi) removing virus from a computer network while guaranteeing e-mail or message can still be received by each person; (vii) removing virus from a computer network while guaranteeing e-mail or message can still be received by each person.

We provide an explicit rule to implement it distributively while also taking into account the network structure information prevents applicability of the proposed distributed algorithm. It is shown from both analysis on complete graphs and simulations on random networks that the proposed strategy tends to balance degree distribution by reducing the gap between maximum and minimum degrees of the network. This topological insight may be used to reduce search space when solving the combinatorial problem in the presence of global network topology information.

We investigate topological properties on solution to the proposed distributed algorithm. It is shown from both analysis on complete graphs and simulations on random networks that the proposed strategy tends to balance degree distribution by reducing the gap between maximum and minimum degrees of the network. This topological insight may be used to reduce search space when solving the combinatorial problem in the presence of global network topology information.

We compare performance of the proposed distributed design method, the brute-force search and random removal strategy for different number of link removals via simulations on several random networks.

The organization of this paper is as follows: basic notions from graph theory and well-known results which will be used to develop the algorithm are presented in Section 2. After formally stating the problem in Section 3, the proposed connectivity-preserving distributed algorithms to remove some links from a network are described in Section 4. Moreover, its optimality and topological insights are discussed in Section 5. Finally, the proposed strategy is demonstrated and its performance is evaluated via several numerical examples in Section 6.

2 Notation and Preliminaries

In this section, we recall some basic notions from graph theory and well-known results which will be used for the development of distributed link removal strategy.

2.1 Notation

Let \( \mathbb{R} \) be the set of real numbers and vector \( 1_n \in \mathbb{R}^n \) denote the column vector of all ones. Furthermore, \( \text{diag}(a) \in \mathbb{R}^{n \times n} \) represents the diagonal matrix with the elements of vector \( a \in \mathbb{R}^n \) on its diagonal. For a given set \( \mathcal{N} \), \( |\mathcal{N}| \) denotes the number of the elements in this set. For a symmetric matrix \( C \), let \( \lambda_i(C) \) denote its eigenvalues which without loss of any generality is ordered as

\[
\lambda_1(C) \leq \lambda_2(C) \leq \ldots \leq \lambda_n(C) = \lambda_{\text{max}}(C)
\]

and its spectral radius is defined as \( \rho(C) \triangleq \max_i(|\lambda_i(C)|) \).

Let \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \) be an undirected graph with a set of vertices \( \mathcal{V} = \{1, 2, \ldots, n\} \) and a set of edges (links) \( \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \). Hence, if \( (i, j) \in \mathcal{E} \), then \( j, i \in \mathcal{E} \) and nodes \( i \) and \( j \) are neighbors. The neighborhood set of node \( i \) is denoted by \( \mathcal{N}_i \subseteq \mathcal{V} \). The degree matrix of a graph, \( \mathcal{D}(\mathcal{G}) \in \mathbb{R}^{n \times n} \), is given by \( \mathcal{D}(\mathcal{G}) \triangleq \text{diag}(d) \),
where \( d = [d_1, \ldots, d_n]^T \) and \( d_i = |A|_i \). The adjacency matrix of a graph, \( \mathcal{A}(G) \in \mathbb{R}^{n \times n} \), is defined as

\[
[\mathcal{A}(G)]_{ij} = \begin{cases} 1, & \text{if } (i, j) \in \mathcal{E}, \\ 0, & \text{otherwise}. \end{cases}
\]

The Laplacian matrix of a graph \( G, \mathcal{L}(G) \in \mathbb{R}^{n \times n} \) is then given by

\[
\mathcal{L}(G) = \mathcal{A}(G) - \mathcal{D}(G).
\]

A graph is connected if there is a path between any pair of distinct nodes. When a graph \( G \) is connected, the eigenvalues of its Laplacian satisfy \( \lambda_1(\mathcal{L}(G)) = 0 \) and \( \lambda_2(\mathcal{L}(G)) > 0 \). For graph \( G \), we denote \( l_G \in \{1, \ldots, |\mathcal{E}|\} \) the index of the edges in \( G \).

Matrix \( C \) is nonnegative if all its elements are nonnegative. Nonnegative symmetric matrix \( C \) is irreducible if and only if its associated graph \( G \) is connected. Nonnegative symmetric matrix \( C \) is said to be primitive if there exists a positive integer \( k \) such that \( C^k > 0 \).

A sufficient condition for matrix \( C \) to be primitive is that it is irreducible and has at least one positive diagonal element [18]. For a primitive matrix \( C \), its largest eigenvalue \( \lambda_{\text{max}}(C) \) satisfies \( \lambda_{\text{max}}(C) > |C_i| \) for \( i \neq n \) and thus we have \( \rho(C) = \lambda_{\text{max}}(C) \) [19].

### 2.2 Max-Consensus Algorithm

Consider an undirected graph \( G \) with \( n \) nodes and let us assign state \( x_i(t) \in \mathbb{R} \) to each node of \( G \). The max-consensus algorithm allows all nodes to compute distributively the maximum among the initial conditions \( x_i(0) \) for \( i = 1, \ldots, n \). Specifically, each node executes the following update rule [20]

\[
x_i(t + 1) = \max_{j \in N_i(t)} x_j(t). \tag{2}
\]

Algorithm (2) can also be modified in a straightforward manner to compute the minimum among the initial conditions \( x_i(0) \), i.e., min-consensus protocol. It is known that update rule (2) makes all nodes \( x_i(t) \) converge to \( \max x_i(0) \) in no more than \( n \) steps, i.e., its computational complexity is equal to \( O(n) \). It will be demonstrated throughout the paper that max-consensus algorithm serves as a unified framework to solve our problem.

### 2.3 Matrix Perturbation

In this paper we will utilize the following result on matrix perturbation [21, p.183].

**Lemma 1.** Consider a symmetric matrix \( A \). Let \( \lambda \) be a simple eigenvalue of \( A \) with eigenvector equal to \( \nu \) and let \( \overline{A} = A + \Delta A \) be a perturbation of \( A \). Then, there is a unique eigenvalue \( \overline{\lambda} \) of \( \overline{A} \) such that

\[
\overline{\lambda} = \lambda + \frac{\nu^T \Delta A \nu}{\nu^T \nu} + O(||\Delta A||^2). \tag{3}
\]

### 2.4 Distributed Power Iteration Method

Consider a primitive symmetric matrix \( P \in \mathbb{R}^{n \times n} \), Its eigenvector \( \nu_n = [\nu_{n1}, \cdots, \nu_{nn}]^T \) corresponding to the simple largest eigenvalue \( \lambda_{\text{max}}(P) \) can be estimated using power method by performing the following iterations [22]

\[
\hat{\nu}_n(t + 1) = \frac{P\hat{\nu}_n(t)}{||P\hat{\nu}_n(t)||_\infty}, \tag{4}
\]

where \( \hat{\nu}_n(t) \) is the estimate of \( \nu_n \) at the \( t \)-th iteration. Since \( \nu_n > 0 \), any choice of initial condition \( \hat{\nu}_n(0) > 0 \) satisfies \( \hat{\nu}_n(0) \cdot \nu_n \neq 0 \) guarantees that \( \hat{\nu}_n \) will asymptotically converge to \( \nu_n \) with rate of convergence equal to \( \left| \frac{\lambda^+(P)}{\lambda_{\text{max}}(P)} \right|^2 \) where \( \lambda^+ \) denotes the second largest eigenvalue of \( P \) in magnitude. Since the graph \( G \) associated with \( P \) is connected, the norm \( ||P\hat{\nu}_n(t)||_\infty \) can be computed distributively via max-consensus algorithm (2) by setting \( x_i(t) = \sum_{j \in N_i(t)} |P_{ij}| \nu_{ij}(t) \). Hence the iteration (4) can be performed in a distributed fashion. In this paper, we adopt the following distributed stopping condition for iteration (4) [23]

\[
\sqrt{n}||r(t)||_\infty \leq \bar{\varepsilon} \left( \min_{i \in G} \hat{h}_i(t) \right), \tag{5}
\]

for a given threshold \( \bar{\varepsilon} \in [0, 1) \) where

\[
r(t) \triangleq \max_{i \in G} \hat{h}_i(t) - P\hat{\nu}_n(t),
\]

\[
\hat{h}_i(t) \triangleq \frac{1}{\nu_i(t)} \sum_{j \in N_i(t)} |P_{ij}| \nu_{ij}(t).
\]

Similarly, condition (5) can also be checked via max/min-consensus protocol (2). At the end of distributed estimation process, node \( i \) will have the estimate of \( \nu_{i,j} \). At each iteration \( t \), each node requires at most \( 4n \) steps to compute \( \nu_{i,j} \) since \( n \) steps are required to compute \( ||P\nu_{i,j}||_\infty \) and a total of \( 3m \) steps are necessary to check stopping condition (5). Given threshold \( \bar{\varepsilon} \), the estimation error of \( \nu_n \) is bounded by [23]

\[
||\nu_n - \hat{\nu}_n(t)||_\infty \leq 2\bar{\varepsilon} \left( 1 + \frac{\bar{\varepsilon}}{1 - \bar{\varepsilon}} \right) \frac{\lambda_{\text{max}}(P)}{\lambda_{\text{max}}(P) - \lambda_{\text{max}}^{-1}(P)}. \tag{6}
\]

**Remark 1.** In general the (distributed) power iteration method, e.g. [24], [25], [26] converges asymptotically. The work [27] presents distributed algorithm to estimate in finite-time the eigenvector corresponding to the spectral radius of a matrix. However, the result is only limited to matrix whose spectral radius is equal to one and thus is not applicable to our problem.

**Remark 2.** The reason we perform normalization in (4) is to overcome the problem as the iteration numbers increase. Specifically, the nonzero components in the iteration vector may become extremely large values if \( \lambda_{\text{max}} > 1 \) or can approach zero if \( \lambda_{\text{max}} < 1 \). Since normalization has no effect on the convergence property of the power method [28, sec. 7.4.1], it can then be done intermittently.

### 3 Problem Formulation

Consider an \( n \) node network whose connections can be represented by an (unweighted) undirected graph \( G_0 = \{V, \mathcal{E}_0\} \). In addition, let us assume that the graph \( G_0 \) is connected and our goal is to remove at most \( m_e \) number of links \( \Delta \mathcal{E} \) from \( \mathcal{E}_0 \) such that the largest eigenvalue of adjacency matrix of the resulting graph \( G_{m_e} = \{V, \mathcal{E}_0 \setminus \Delta \mathcal{E} \} \) is minimized. As a motivating application, it is well-known that the largest eigenvalue \( \lambda_{\text{max}} \) is the key to spreading of disease under various types of epidemic models [7], [29], [30]. Specifically, the epidemics die out if \( \lambda_{\text{max}} \leq \beta_e \) where \( \beta_e \) is a threshold that depends on the epidemic models. Hence, one method to control the epidemic is by removing some links from a network to reduce its largest eigenvalue. The problem in general can then be mathematically formulated as the following optimization problem:

\[
\begin{align}
\min_{\Delta \mathcal{E}} \quad & \lambda_{\text{max}}(\mathcal{A}(\mathcal{G}_{m_e})), \\
\text{s.t.} \quad & |\Delta \mathcal{E}| \leq m_e, \\
& \Delta \mathcal{E} \subseteq \mathcal{E}_0, \\
& \lambda_2(\mathcal{L}(\mathcal{G}_{m_e})) > 0,
\end{align}
\]
where the last constraint implies that the resulting topology is connected. Using the labeling \( l_{\theta_{i}} \in \{1, \ldots, |\theta|\} \) on graph \( \theta \) introduced in the previous subsection, the optimization problem (P1a) can then be restated as:

\[
\begin{align*}
\min_{\hat{y}} & \quad \lambda_{\text{max}} (\mathcal{A}(\theta_{0}) - \Delta \mathcal{A}^{-}) , \\
\text{s.t.} & \quad \Delta \mathcal{A}^{-} = \sum_{l_{\theta_{i}} = 1}^{\theta} \tilde{y}_{l_{\theta_{i}}} \mathcal{A}_{l_{\theta_{i}}} , \\
& \quad \lambda_{2} (\mathcal{L}(\theta_{m_{e}})) > 0 , \\
& \quad \sum_{l_{\theta_{i}} = 1}^{\theta} \tilde{y}_{l_{\theta_{i}}} \leq m_{e} , \\
& \quad \tilde{y} \in \{0, 1\}^{\theta} 
\end{align*}
\]  

(P1b)

where \( \mathcal{A}_{l_{\theta_{i}}} \) is a matrix with all zeros entries except for the \( ij \)-th and \( ji \)-th entries corresponding to the edge of label \( l_{\theta_{i}} \) which are equal to 1, and the vector \( \tilde{y} = [\tilde{y}_{1}, \ldots, \tilde{y}_{|\theta|}]^{T} \) is a Boolean vector, i.e., \( \tilde{y}_{l_{\theta_{i}}} \in \{0, 1\} \) where \( \tilde{y}_{l_{\theta_{i}}} = 1 \) means that the edge \( l_{\theta_{i}} \) in \( \theta \) is removed.

Remark 3. The formulation in (P1b) can also be modified to take into account the degradation of the convergence speed to consensus as will be shown later in this paper. Note that the optimization (P1b) can be solved before the network is employed or after the employment since its topology may change.

The optimization problem (P1b) is a combinatorial one whose global solution can be computed using brute-force search. However, the complexity increases exponentially with the network size. In addition, one common issue in solving (P1b), including its relaxation, for example in [3], is that the global knowledge on network topology \( \theta \) is required or assumed to be known. However, in reality such information is often not available due to, e.g., privacy reasons and geographical constraint or hard to obtain due to the size of the network. Hence, in this paper we impose the following constraint.

Constraint 1. The overall network topology \( \theta_{0} \) is not available and each node \( i \) only knows the information on \( \mathcal{N}_{i} \).

The absence of information on global network topology makes it impossible to solve (P1b) in a centralized manner. Therefore, in this paper we propose a distributed strategy performed by each node to solve (P1b) using only its local information on the network structure given by \( \mathcal{N}_{i} \).

Problem 1. Assume that \( \theta_{0} \) is connected. Find a suboptimal solution or an upper bound to the solution to optimization (P1b) under constraint 1 and by solely using local information on \( \mathcal{N}_{i} \).

In this paper, for the sake of simplicity it is assumed that the nodes know an upper bound on the size of the network, denoted by \( \bar{\theta} \). Otherwise, the value can be estimated distributively using existing methods such as the ones presented in [31], [32].

4 DISTRIBUTED LINK REMOVAL ALGORITHM WITH CONNECTIVITY PRESERVATION

In this section, we formulate an optimization problem whose solution is equal to a suboptimal solution to (P1b) and describe a method to solve it in a distributed manner.

4.1 A Suboptimal Solution to Link Removal Problem

Based on the matrix perturbation analysis presented in Section 2.3, the matrix \( \Delta \mathcal{A}^{-} \) in (P1b) can be treated as a perturbation added to the matrix \( \mathcal{A}(\theta_{0}) \). It is worth to note that perturbation analysis has also been used to study link addition [33] and node removal problems [34]. First, observe that since \( \theta_{0} \) is connected, then matrix \( \mathcal{A}(\theta_{0}) \) is nonnegative and irreducible. This implies that eigenvalue \( \lambda_{\text{max}} (\mathcal{A}(\theta_{0})) \) is simple [18, Theorem 8.4.4]. Using Lemma 1 and considering the matrix \( \bar{\mathcal{A}} = \mathcal{A}(\theta_{0}) + \Delta \mathcal{A}^{-} \) with \( \Delta \mathcal{A} = \Delta \mathcal{A}^{-} \), the following lemma and corollary, whose proofs can be found in the appendices, provide condition so that

\[
\bar{T}_{\text{max}} = \lambda_{\text{max}} (\mathcal{A}(\theta_{0})) + \frac{\nu^{T} \Delta \mathcal{A} \nu}{\nu^{T} \nu} + O (\|\Delta \mathcal{A}^{2}\|)
\]  

(7)

where \( \bar{T}_{\text{max}} \) is the largest eigenvalue of \( \bar{\mathcal{A}} \) and \( \nu \) is the eigenvector associated with \( \lambda_{\text{max}} (\mathcal{A}(\theta_{0})) \).

Lemma 2. For a given number of link removals \( m_{e} \), assume that the following condition on graph \( \theta_{0} \) is satisfied:

\[
|\lambda_{\text{max}} (\mathcal{A}(\theta_{0})) - \lambda_{n-1} (\mathcal{A}(\theta_{0}))| > 2 \sqrt{2m_{e}} .
\]  

Then, Equation (7) holds.

For a single link removal, i.e. \( m_{e} = 1 \), we can compute \( \|\Delta \mathcal{A}^{-}\|^{2} = \|\Delta \mathcal{A}_{l_{\theta_{i}}}\|^{2} \) explicitly and thus improve the bound in (8).

Corollary 1. For \( m_{e} = 1 \), the condition (8) is given by

\[
|\lambda_{\text{max}} (\mathcal{A}(\theta_{0})) - \lambda_{n-1} (\mathcal{A}(\theta_{0}))| > 2.
\]  

Note that the higher order term in (7) depends on the eigenvalues and eigenvectors of adjacency matrix \( \mathcal{A}(\theta_{0}) \) as can be seen in Section 5.1 and can be estimated/computed in a distributed manner, for example using the method proposed in [35]. In the rest of the paper, for simplicity we neglect the higher order term in (7). Under assumption (8), from (7) with \( \Delta \mathcal{A} = -\Delta \mathcal{A}^{-} \) and since deleting a link will always decrease \( \lambda_{\text{max}} \) [36, Proposition 1.3.10] we then have

\[
\bar{T}_{\text{max}} \leq \lambda_{\text{max}} (\mathcal{A}(\theta_{0})) + \Delta \lambda_{\text{max}}
\]

where \( \Delta \lambda_{\text{max}} = \frac{\nu^{T} \Delta \mathcal{A} \nu}{\nu^{T} \nu} \) and can be computed as

\[
\Delta \lambda_{\text{max}} = \frac{\nu^{T}}{\nu^{T}} \left( \eta_{l_{\theta_{i}}} \tilde{y}_{l_{\theta_{i}}} \mathcal{A}_{l_{\theta_{i}}} \right) \nu
\]

\[
= - \frac{2}{\nu^{T} \nu} \sum_{l_{\theta_{i}} = 1}^{\theta} \tilde{y}_{l_{\theta_{i}}} \nu_{i} \nu_{j} = - \Delta \bar{\lambda}_{\text{max}}.
\]

Note that since \( \mathcal{A}(\theta_{0}) \) is non-negative and irreducible matrix, we have \( \nu_{i} > 0 \) for all \( i \) which yields \( \Delta \bar{\lambda}_{\text{max}} > 0 \). Therefore we can write

\[
\min_{\tilde{y}} \bar{T}_{\text{max}} \leq \lambda_{\text{max}} (\mathcal{A}(\theta_{0})) + \min_{\tilde{y}} \left( -\Delta \bar{\lambda}_{\text{max}} \right)
\]

\[
\leq \lambda_{\text{max}} (\mathcal{A}(\theta_{0})) - \max_{\tilde{y}} \Delta \bar{\lambda}_{\text{max}}.
\]

Hence, a suboptimal solution to (P1b) can be obtained by solving the following optimization problem

\[
\begin{align*}
\max_{\tilde{y}} & \quad \Delta \bar{\lambda}_{\text{max}} \\
\text{s.t.} & \quad \lambda_{2} (\mathcal{L}(\theta_{m_{e}})) > 0 , \\
& \quad \sum_{l_{\theta_{i}} = 1}^{\theta} \tilde{y}_{l_{\theta_{i}}} \leq m_{e} , \\
& \quad \tilde{y} \in \{0, 1\}^{\theta} 
\end{align*}
\]  

(P1c)

where

\[
\Delta \bar{\lambda}_{\text{max}} = \frac{2}{\nu^{T} \nu} \sum_{l_{\theta_{i}} = 1}^{\theta} \tilde{y}_{l_{\theta_{i}}} \nu_{i} \nu_{j},
\]  

(9)
4.2 Choosing Threshold $\tau$ for Distributed Power Iteration Method

In order to solve optimization problem (P1c) distributively, we utilize distributed power method in Section 2.4 to distributively estimate $\hat{v}_n$. To this end, we define the primitive matrix $P$ in (4) as

$$ P = I_n + \mathcal{A}(\mathcal{G}_0). \tag{10} $$

It is clear that the eigenvalues of $P$ and $\mathcal{A}(\mathcal{G}_0)$ satisfy $\lambda(P) = 1 + \lambda_i(\mathcal{A}(\mathcal{G}_0))$ and they share the same set of eigenvectors. Since matrix $P$ is primitive, we have $\rho(P) = \lambda_{\text{max}}(P)$ and thus $\rho(P) = 1 + \lambda_{\text{max}}(\mathcal{A}(\mathcal{G}_0))$. Moreover, since $\mathcal{G}_0$ is connected, the eigenvalue $\lambda_{\text{max}}(P)$ is simple and its corresponding eigenvector has positive elements.

Since power iteration method converges asymptotically, in the following we provide a guideline for choosing threshold $\tau$ in (5) which guarantees the solution to (P1c) based on the estimated eigenvector $\hat{v}_n$, are equal to the one using the true eigenvector $v_n$.

**Lemma 3.** For connected graph $\mathcal{G}_0$ and a maximum number of links to be removed $m$, let the threshold $\tau$ in (5) be chosen as

$$ \tau < \frac{\sqrt{\lambda_{\text{max}}(\mathcal{A}(\mathcal{G})) + \lambda_{\text{max}}(\mathcal{A}(\mathcal{G}_0))}}{4 + \frac{1}{\gamma} + \frac{1}{\sqrt{\lambda_{\text{max}}(\mathcal{A}(\mathcal{G}))}}} \tag{11} $$

where $\lambda_{\text{max}}(\mathcal{G})$ is the maximum degree of $\mathcal{G}_0$.

In order to compute (11), the nodes start with a sufficiently small $\tau$ to obtain $\hat{v}_n$ and then improve the estimates by reducing $\tau$ such that condition in (11) is satisfied. While $\lambda_{\text{max}}$ can be computed distributively using max-consensus protocol in (2) with $x_i(0) = d_i$, we only need information on the lower bound of the spectral gap for checking condition (11). If the spectral gap is known to be sufficiently large, we can replace $\lambda_{\text{max}}(\mathcal{A}(\mathcal{G})) - \lambda_{\text{max}}(\mathcal{A}(\mathcal{G}_0))$ in (11) with a sufficiently small value, e.g., 0.0001. The set $\mathcal{E}_m = \{\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_{m-1}\}$ is the set of links with $(m+1)$ largest $\hat{v}_n, \hat{v}_j$. We then have

$$ \text{argmax}_{l \in \mathcal{E}_0} \hat{v}_n, l \hat{v}_j = \text{argmax}_{l \in \mathcal{E}_0} \hat{v}_n, l \hat{v}_j $$

with $P^m(\mathcal{G}) \leq m$.

**Proof.** See Appendix C.

4.3 Distributed Connectivity Verification

Let the link $(i^*, j^*)$ be given by the solution to

$$ (i^*, j^*) = \text{argmax}_{(i,j) \in \mathcal{E}} \hat{v}_n, l \hat{v}_j. \tag{14} $$

In the following we utilize the max-consensus protocol (2) to check distributively connectivity of graph $\mathcal{G}_0 = (\mathcal{G}_0 \setminus \{i^*, j^*\})$. To this end, we first set initial values in (2) as $x_i(0) = 0, \forall i \in \mathcal{G}$. After computing the candidate link to be removed $(i^*, j^*)$, either node $i^*$ or $j^*$ modifies its initial value to $x_i(0) = 1$ or $x_j(0) = 1$ (nodes $i^*$, $j^*$ can either locally negotiate, e.g. based on the estimate $\hat{v}_n, l \hat{v}_j$, or randomly choose, e.g. by flipping a coin, among themselves to decide which one modifies its initial value to one), while the remaining nodes keep their initial values unchanged. Without loss of generality, assume that $x_i(0) = 1$ and $x_j(0) = 0, \forall j \neq i^*$. All nodes then execute (2) on graph $\mathcal{G}_0 = (\mathcal{G}_0 \setminus \{i^*, j^*\})$, which means that

$$ \forall n \in \mathcal{G}_0, (n, i^*) \notin \mathcal{E} \Rightarrow x_i(0) = 1. $$

Fig. 1: Methods for computing a suboptimal solution to (P1b) using iterative and simultaneous link removal strategies

and $\mathcal{v}_{n,i}$ denote the $i$-th element of the eigenvector $\mathcal{v}_n$. **Remark 4.** Condition (8) can be checked in a distributed manner using the method in [37] or [35]. Note that network with a large spectral gap has been studied in the literature, see e.g., [38]. It is known that good expander graphs have large spectral gap [39, p. 182]. For example, the spectral gap of a complete graph with $n$ nodes is equal to $n$. As can be seen from the previous discussion, when condition (8) is satisfied, it is sufficient to focus only on the movement of $\lambda_{\text{max}}$ in order to predict or estimate $\hat{\lambda}_{\text{max}}$. In addition, large spectral gap also allows us to neglect the high order term in (7) [21, p.185] while obtaining a suboptimal solution close to the global one as also indicated by the simulation results in Section 6.1.

When condition (8) is not satisfied or the spectral gap $(\lambda_{\text{max}} - \lambda_{\text{min}} - 1)$ is small, distributed strategy proposed in the paper can still be used to compute a suboptimal solution and the solution can be improved by iteratively removing one (or some) link at a time from the graph $\mathcal{G}_0$ for a total maximum $m$ iterations as shown in Section 6.1. This iterative strategy can also be seen as a greedy algorithm for solving optimization problem (P1b) as illustrated in Fig. 1. In addition, by iteratively removing one link at a time, the error introduced by the matrix perturbation will be minimized at each iteration. The iterative strategy is also suitable for some classes of network with $\mathcal{v}_{n,i} = \mathcal{v}_{n,j}$ for all $i \neq j$ such as in regular graph, where the first-order perturbation cannot predict the movement of eigenvalue $\lambda_{\text{max}}$. Another strategy is by incrementally reducing the weight of the link, namely the matrix $\Delta \mathcal{A}$ in (7) is given by $\varepsilon \Delta \mathcal{A}$ as shown in Section 6.1. By choosing $\varepsilon \ll 1$, equation (7) will still hold since the perturbation matrix $\Delta \mathcal{A}$ is sufficiently small.
node \(i^*\) does not use the information it received from node \(j^*\) in its consensus protocol and vice versa. If the resulting graph \(G_k\) is connected, then after \(\pi\) steps we have \(x_i(\pi) = 1, \forall i \in \mathcal{V}'\). However, when the removal of link \((i^*, j^*)\) results in disconnected graph \(G_{k}^{'}\), then \(x_j(\pi) = 0\). To be more precise, the nodes are partitioned into two groups denoted by \(Y_1\) and \(Y_2\) where, with no loss of generality, \(i^* \in Y_1\) and \(j^* \in Y_2\) whose states are equal to \(x_i(\pi) = 1, i \in Y_1\) and \(x_j(\pi) = 0, j \in Y_2\) respectively. Hence, the pair of nodes \((i^*, j^*)\) can estimate connectivity of the resulting network by checking whether \(x_i(\pi) = x_j(\pi)\) and broadcast the result to the remaining nodes in the network using maximum consensus protocol. The pseudo-code for distributed connectivity detection is shown in Algorithm 1.

Algorithm 1 can also be used to check connectivity of the resulting graph when multiple links are being removed simultaneously. To be more precise, the set of links to be removed is given by \(\delta^c \subset \delta_0\) with \(|\delta^c| = m_c > 1\). The initial states of nodes are chosen similar to the case of a single link removal (i.e., only one node in the set \(\delta^c\)) with initial value equal to one while the rest are zero) and all nodes execute max-consensus protocol (2) on graph \(G_m\) = \((\mathcal{V}', \delta_0 \setminus \delta^c)\). If there exists at least one pair of nodes \((i^*, j^*)\) in \(\delta^c\) that after \(\pi\) steps \(x_i(\pi) \neq x_j(\pi)\), which can be checked locally, it can then be concluded that the resulting graph \(G_m\) is disconnected.

### 4.4 The Complete Distributed Link Removal Algorithm

The pseudo-code of the distributed algorithms to compute a suboptimal solution to (P1b) by removing a single link (resp. multiple links) iteratively is summarized in Algorithm 2 (resp. Algorithm 3). Note that removing multiple links may result in faster computation since the number of iterations in Algorithm 2 is less than \(m_c\). However, the convergence may degrade in comparison to the removal of one link at a time as will be shown in Section 6. Furthermore, by setting \(m = m_c\), Algorithm 3 can also be applied to remove \(m_c\) links simultaneously. Both Algorithms 2 and 3 utilize (distributed) power iteration and max-consensus algorithms. As discussed in Section 2, the convergence rate of power iteration is equal to \(|\lambda^*(P)/\lambda_{\text{max}}(P)|^2\) where \(P\) is given in (10) while the complexity of max-consensus algorithm is \(O(n)\). Moreover, note that the max-consensus protocol (2) is used at each iteration of the power method (namely for computing \(|P_{\delta_0}^{(t)}|\) and checking stopping condition (5)) and also for computing (9) and checking the connectivity (Algorithm 1).

In the following we provide some remarks on the proposed distributed algorithms.

(i) It is worth noting that initial values used to estimate the eigenvector and to check the connectivity are not the true initial values of the nodes and thus do not contain any physical meaning. Therefore, the proposed algorithm preserves privacy of nodes in a network in terms of their true initial values and does not collide with the privacy preserving consensus, e.g., the one proposed in [42].

(ii) The set of links that disconnect the resulting network obtained at each iteration in Algorithm 2 can be saved and those links will be excluded in the next iteration to shorten...
the time required for checking the connectivity of the resulting network. In addition, for a large size network, in many situations its eigenvector does not change much by a small perturbation (i.e., single link removal) [43]. Hence, the estimated eigenvector at the current iteration $t$ can be used as an initial value $\hat{y}_n(0)$ for the next iteration $t + 1$ to speed up the convergence of power method.

(iii) When the initial network $G_0$ is disconnected, then algorithms 2 or 3 can be applied in parallel to all subnetworks.

(iv) As shown in Algorithms 2 and 3, the proposed approach is not only able to find a suboptimal solution to (P1b) distributively but also to avoid the combinatorial issue without applying any relaxation to the Boolean variable.

(v) When graph $G_0$ is given by a weighted undirected graph, Equation (9) can then be written as

$$\Delta \lambda_{\text{max}} = \frac{2}{\nu_n \nu_n \sum_{i,j=1}^{|V_n|} \hat{y}_n(i)w_{ij}y_{n,i}y_{n,j}}$$

where $w_{ij}$ denote the weight of link $(i, j)$. Hence, the algorithms 2 and 3 can be extended in a straightforward manner to solve (P1c) in a distributed manner with $\Delta \lambda_{\text{max}}$ is defined in (15) and provided that each node $i$ knows $w_{ij}$ for $j \in N_i$.

(vi) The proposed distributed algorithm can also be extended when desired critical value $\beta^*$ is given and the following additional constraint in (P1b) is required to be satisfied 3]

$$\lambda_{\text{max}}(\mathcal{L}(G)) < \beta^*$.

(vii) The link removal implies that the speed of convergences to consensus, defined by $\lambda_2(\mathcal{L})$ be reduced. In order to study the trade-off between the speed for reducing the spread of disease and the convergence speed to consensus, we can formulate and solve the following optimization problem:

$$\min_{\Delta \phi^-} \left(1 - \zeta \right) \lambda_{\text{max}}(\mathcal{L}(G(\phi))) - \zeta \lambda_2(\mathcal{L}(G))$$

s.t. $|\Delta \phi^-| \leq m_\phi$, $\Delta \phi^- \subseteq \phi_0$,

$$\lambda_2(\mathcal{L}(G)) > 0,$$

where $\zeta \in [0, 1]$ is a constant representing the trade-off between the two metrics. Similarly, the value of $\lambda_2(\mathcal{L}(G))$ can be expressed as $\lambda_2(\mathcal{L}(G(\phi)) = \lambda_2(\mathcal{L}(G_0)) + \Delta \phi$. Note that $\lambda_2$ can also be approximated similarly in a distributed manner [2], [41].

(viii) The proposed distributed strategy can also be applied to link addition problem whose goal is to maximize $\lambda_{\text{max}}(\mathcal{L}(G))$ (e.g., maximizing the spread of information in a network). Similarly, by using perturbation analysis as presented in Section 4.1 the problem can be approximated as to maximize $\Delta \lambda_{\text{max}}$ and thus, the strategy described in Section 4.2 can be utilized to compute a suboptimal solution. Another possible objective is to maximize the algebraic connectivity, i.e., $\lambda_2(\mathcal{L})$ as discussed in [41].

5 Optimality and Topological Insight

This section discusses the optimality of the proposed approach in comparison to the brute-force search in the presence of global network topology information. Moreover, the topological insight of the proposed algorithm is also investigated.

5.1 Optimality of the Solution

In the following, we analyze the error between the solutions to (P1b) and (P1c). First, we discuss the optimality gap between the solution to (P1c) based on the simultaneous link removal strategy and the global solution to (P1b). Assume that condition (8) is satisfied and let the solution to optimization (P1c) for the simultaneous removal of $m_\phi$ links be given by $\phi^*$. The exact largest eigenvalue of the perturbed matrix by removing the links $\phi^*$ can be computed as [21]

$$\tilde{\lambda}_{\text{max}}^\phi = \lambda_{\text{max}} - \sum_{(i', j') \in \phi^*} 2v_{i', j'}v_{i', j'} + \frac{\sum_{j=1}^{n-1} \left[ \sum_{(i', j') \in \phi^*} (v_{i', j'}v_{i', j'} + v_{i, j'}v_{i', j'}) \right]^2}{\lambda_{\text{max}} - \lambda_j} + O(\|A - \hat{A}\|^3).$$

Similarly, let the solution to the original optimization problem (P1b) be given by the set $\mathcal{E}$. The exact largest eigenvalue of the perturbed matrix by removing the links $\mathcal{E}$ is then given by

$$\tilde{\lambda}_{\text{max}} = \lambda_{\text{max}} - \sum_{(i, j) \in \mathcal{E}} 2v_{i, j}v_{i, j} + \frac{\sum_{j=1}^{n-1} \left[ \sum_{(i, j) \in \mathcal{E}} (v_{i, j}v_{i, j} + v_{i, j}v_{i, j}) \right]^2}{\lambda_{\text{max}} - \lambda_j} + O(\|A - \hat{A}\|^3).$$

Therefore, the optimality gap between the solutions to (P1b) and (P1c), i.e., $\tilde{\lambda}_{\text{max}} - \lambda_{\text{max}}^\phi = \lambda_{\text{max}} - \lambda_{\text{max}}$ is equal to the difference between (17) and (16). As can be observed, the gap depends on all eigenvalues together with the associated eigenvectors of adjacency matrix $\mathcal{L}(\phi_0)$. Note that for a network with large spectral gap ($\lambda_{\text{max}} - \lambda_1$) and $j \neq n$, the second and higher terms in (16) and (17) become small and thus can be neglected. Hence, in this case the gap depends solely on the eigenvector associated with $\lambda_{\text{max}}$. This also indicates that the gap between (P1c) and (P1b) is small.

Next, we discuss the gap between the solution obtained from Algorithm 2 and the global solution to (P1b). As can be seen from Fig. 1, the gap is equal to the summation of the error from the greedy algorithm and the perturbation based approximation. It is unknown if there is a bound on the error between the solution to (P1b) and the one based on greedy algorithm unless the function $\lambda_{\text{max}}(\mathcal{L}(\phi_0))$ shares a similar property to supermodular function of the edges, which to the best of the authors’ knowledge, is still an open problem. The gap between the greedy based algorithm and its perturbation based approximation is given by the differences between (16) and (17). From the simulations presented in Section 6.1, it is observed that when the spectral gap ($\lambda_{\text{max}} - \lambda_{\text{max}} - \lambda_n$) is large, the gap between the solutions is quite small. In addition, it is worth noting that the value of

$$\tilde{\lambda}_{\text{max}}^\phi = \lambda_{\text{max}} - \sum_{(i', j') \in \phi^*} 2v_{i', j'}v_{i', j'}$$

for the simultaneous multiple link removal strategy is larger than single link removal strategy, meaning that the first-order approximation poorly approximates the exact eigenvalue of the perturbed matrix for the simultaneous multiple link removal case. This is one of the reasons why in general removing iteratively one link at a time yields a better solution compared to simultaneous link removal strategy as mentioned in Section 4.1.
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/TNSE.2018.2813426, IEEE Transactions on Network Science and Engineering

5.2 Topological Insights into the Proposed Strategy

In the following we provide topological insights into Algorithm 2. We start by introducing the degree’s gap $\delta$ of a network which is defined as

$$\delta = \max_i d_i - \min_j d_j.$$

(18)

We then have the following as a result of applying Algorithm 2 to a complete graph whose proof can be found in Appendix D.

**Lemma 4.** Consider a complete $n$-node network of degree $(n-1)$ with $\delta = 0$ and assume that $m_e \leq \left\lfloor \frac{n}{2} \right\rfloor$. Then, the degree’s gap of the resulting network after removing $m_e$ links by applying algorithm 2 satisfies $\delta \leq 1$.

In addition, we perform simulations on 100 randomly generated regular graphs with size of 50 nodes and degree equal to 20. For each graph, we apply Algorithm 2 where the number links to be removed $m_e$ is varied between 1 and 150. Figure 2 shows the average degree’s gap $\delta$ for each $m_e$. As can be observed, the degree’s gap of the resulting graph is bounded by two, i.e., $\delta \leq 2$.

It is shown from Lemma 4 and simulations results in Fig. 2 that Algorithm 2 tends to reduce the degree’s gap or balance the degree distribution of a network, i.e., to make the resulting network to be as regular as possible, which supports the claim in [1]. We will also show later in Section 6 that a similar observation can also be found in several random networks and also real network such as power grid.

Note that it is shown in [44] that the difference between the eigenvector $v_n$ of a graph and its degree vector to be the smallest (i.e., $q^T q = 0$ with $q = v_n - \frac{d}{\sqrt{n}}$) when $\lambda_{max} = W_2/W_1$ where $W_k$ denotes the total number of $k$ hop walks between any two nodes which can be the same. In other words, for this type of graph the element of the eigenvector $v_n$ is "proportional" to the degree of node. Therefore, for network with $\lambda_{max} = W_2/W_1$ (e.g., large size Erdös-Rényi random graphs), the proposed strategy in Algorithm 2 is equivalent to removing links whose corresponding nodes have the largest degree which is similar to the strategy presented in [16].

The observation in Lemma 4 does not always hold when multiple links are removed at a time as given by algorithm 3 which will be shown later in Section 6. The observation that Algorithm 2 tends to reduce the degree’s gap may be used for reducing the search space of the original optimization problem (P1b).

6. NUMERICAL EXAMPLES

In this section, we demonstrate and evaluate the proposed distributed strategy via several numerical examples.

6.1 Demonstration and Comparison with Centralized Approach

First, we demonstrate the distributed design procedure and compare it with the centralized approach in the present of global network topology information. We consider a network of 8 nodes whose initial structure $\%0$ is shown in Fig. 3. We choose a small size network so that the comparison with the centralized approach, which in general is NP-hard, becomes possible. The number of links to be removed $m_e$ is varied between 1 and 6. We apply Algorithm 2 in order to remove iteratively a single link at a time. At each iteration $i = \{1, \ldots, m_e\}$, the nodes first estimate $v_n$ distributively. The estimate of $v_n$ graph $\%0$ is given by

$$\hat{v}_n = [0.799, 0.857, 0.786, 0.83, 0.67, 1, 0.779, 0.399]^T.$$

In addition, we also apply Algorithm 3 and modify the number of links removed at each iteration to 2, 3 and 6 links. As a comparison we also solve the original optimization problem (P1b) by performing a brute-force search. Solutions to (P1c) using Algorithms 2, 3 and the optimal solution to (P1b) are shown in Fig. 4. First, it can be observed from Fig. 4 that the Algorithm 2 results in a better performance in comparison to the simultaneous link removal strategy for $m_e = 6$.
Algorithm 4 Iterative link removals based on continuous perturbation matrix

Require: graph $\mathcal{G}_0$ is connected, $m_e, \varepsilon \ll 1$
1: set $\bar{y} = [0, \ldots, 0]^T \in \mathbb{R}^{|\mathcal{G}|}$
2: initialize $A = \mathcal{A}(\mathcal{G}_0)$
3: while $\|A\|_0 > 2(|\mathcal{E}_0| - m_e)$ do
4: compute the optimal link (pair of nodes) $(i^*, j^*) \sim l_{\mathcal{G}_0}^*$ using algorithm 2 for the matrix $A$
5: $A \leftarrow A - \varepsilon A l_{\mathcal{G}_0}^*$
6: end while
7: if $|A|_{ij} > 0$ then
8: $|A|_{ij} = 1$
9: end if
10: if $|\mathcal{A}(\mathcal{G}_0) - A|_{ij} = 1$ then
11: $y_{l_{\mathcal{G}_0}} \leftarrow 1$ where $l_{\mathcal{G}_0} \sim (i, j)$
12: end if
13: the set of links to be removed is given by $\bar{y}$

removal strategy and $m_e = 1, 2, 3$, the solutions to (P1c) are nearly optimal, and in fact equivalent to that of (P1b), i.e., there is no performance loss in spite of the absence of the global network topology. Note that, the value $|\lambda_{\text{max}}(\mathcal{G}(\bar{y}_e))| - |\lambda_{\text{max}}(\mathcal{G}(\bar{y}_e))|$ are equal to 2.6291, 2.2516, 2.0540 for $m_e = 1, 2, 3$, respectively which satisfy (8). However, as $m_e$ increases, the gap between the maximum values of $\lambda_{\text{max}}$ and the solutions to (P1c) and (P1b) becomes larger as the spectral gap becomes smaller. Specifically, we modify Algorithm 2 and apply a simple threshold operation as described in Algorithm 4 to compute a set of links to be removed. The results are summarized in Table 1 in which Algorithm 2 yields a better result compared to Algorithm 4 for $m_e = 1, \ldots, 4$. 

TABLE 1: Comparison of $\lambda_{\text{max}}(\mathcal{A}(\mathcal{G}_{m_e}))$ obtained from Algorithms 2 and 4 for the network depicted in Fig. 3

<table>
<thead>
<tr>
<th>$m_e$</th>
<th>$\Delta^\alpha$</th>
<th>$\mathcal{A}(\mathcal{G})_{\varepsilon = 0}$</th>
<th>$\mathcal{A}(\mathcal{G})_{\varepsilon = 0.0001}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.624</td>
<td>3.657</td>
<td>3.657</td>
</tr>
<tr>
<td>2</td>
<td>3.348</td>
<td>3.497</td>
<td>3.398</td>
</tr>
<tr>
<td>3</td>
<td>3.068</td>
<td>3.244</td>
<td>3.214</td>
</tr>
<tr>
<td>4</td>
<td>2.812</td>
<td>2.948</td>
<td>2.954</td>
</tr>
<tr>
<td>5</td>
<td>2.593</td>
<td>2.659</td>
<td>2.594</td>
</tr>
<tr>
<td>6</td>
<td>2.355</td>
<td>2.302</td>
<td>2.355</td>
</tr>
</tbody>
</table>

Note that the solutions from both Algorithms 2 and 4 are close to the global optimal ones as confirmed from Fig. 4. The computational time required by Algorithm 2 is much less than the one needed by Algorithm 4 in terms of the number of estimation of $y_e$. In addition, it can be observed that a smaller $\varepsilon$ does not necessarily result in a better performance, e.g., for the case of $m_e = 4.6$. This may also indicate that choosing the optimal threshold for Algorithm 4 is not a trivial problem. However, this issue is beyond the scope of the current paper.

6.2 Evaluation on Random Networks

In the following we evaluate the proposed strategy on random networks, namely Barabási-Albert Scale Free model, Gilbert stochastic and Watts-Strogatz models consisting of 200 nodes and with number of links $|\mathcal{E}_0|$ equal to 570, 574 and 639 respectively. These random networks share special properties such as their degree distribution as illustrated in Fig. 5 which may help us to gain insights on the proposed strategy as will be shown later. For each type of random network we first remove a single link using Algorithm 2 where the value $m_e$ is varied between a single link and 30% of $|\mathcal{E}_0|$. In addition, we compare the proposed strategy with random link removal, while preserving the connectivity of the resulting network, as the simplest alternative of distributed strategy. It is clear from Fig. 6 that Algorithm 2 outperforms the random link removal strategy. Moreover, in order to obtain a desired $\lambda_{\text{max}}$, more links are required to be removed when using the random strategy which is undesired since it can deteriorate significantly the speed of the consensus algorithm.

Next, we apply Algorithm 3 by removing 5 links and 20 links at each iteration. As can be observed from Figs. 6a and 6b, for Barabási-Albert Scale Free and Gilbert stochastic network models the resulting value of $\lambda_{\text{max}}$ from removing 5 and 20 links at each iteration are close to the single link removal. This means that for these network models, we can reduce the computational effort of the proposed distributed strategy in terms of the number of iterations without harming the quality of its solutions. However, as can be seen from Fig. 6c, the gap of $\lambda_{\text{max}}$ between a single and multiple link removals at each iteration for the Watts-Strogats model is quite large. Specifically, the performance of Algorithm 2 for 20 link removals at each iteration turns out to be worse than the random link removal strategy. In order to get a hint of the cause for this performance degradation, we investigate the topological property, namely the degree distribution of the resulting network given by Algorithms 2 and 3. To this end, we compute the degree’s gap $\delta$ defined in (18) and the total difference of the degree in the network given by $\sum_{(i \neq j)}|d_i - d_j|$. As can be seen from Figs. 7 and 8, in general lower values of $\delta$ and $\sum_{(i \neq j)}|d_i - d_j|$ yields lower values of $\lambda_{\text{max}}$, i.e., results in a better performance. Furthermore, the solutions given by algorithm 2, i.e., by iteratively removing one link at a time, for all three random networks tend to reduce the degree’s gap $\delta$ which support our analysis in Section 5.2. It is worth noting that for the case of Watts-Strogats network, removing 20 links at each iteration in general yields a larger degree’s gap $\delta$ and total difference of the degree in comparison to the one obtained by the random link removals strategy as depicted in Figs. 7c and 8c. This is one of the reasons why removing 20 links at each iteration results in a worse performance compared to random link removal strategy. Hence, the simulation results suggest and we conjecture that when a network has similar degree distribution to that of regular graph, it is recommended to remove one or small number of links at a time in order to yield good quality of solutions.
that the observations on the degree’s gap and the total difference of the degree might be used as a hint to decide whether a given number of links to be removed at each iteration is a good choice. In addition, this insight may also be combined with the proposed strategy to improve the quality of solutions obtained by removing multiple links at a time for Watts-Strogatz network.

Finally, let us assume that at each iteration $m$ number of links are going to be removed from the networks in Fig. 5 while preserving their connectivity. Fig. 9 shows the number of links involved in the connectivity verification at each iteration for the proposed strategy (with different values of $m$), which are not less than $m$ as can be observed from Algorithm 3. It can be seen that for the case of $m = 1, 5$ the number of links involved in the connectivity verification for all the networks considered in Fig. 9 are equal to $m$. However, for the case of $m = 20$ and Watts-Strogatz network model the number of links involved in the connectivity verification are slightly larger than $m$ while for the scale-free and Gilbert stochastic network models the number are equal to $m$. 

Fig. 5: Degree distribution of Random networks models of 200 nodes considered in the numerical example

Fig. 6: Proposed approach vs. random link removal strategy for random network of 200 nodes. The proposed distributed algorithm outperforms the random link removal strategy for all three random networks. For (a) and (b), multiple link removals at each step yields a performance near the single link removal at each step. On the other hand, for case (c) multiple link removals results in much worse performance compared to the single link removal and may be worse than the random link removal strategy.

Fig. 7: Degree’s gap for different link removal strategies. As can be observed, algorithm 2 tends to reduce the degree gap of the three random networks. On the other hand, algorithm 3 does not always reduce the degree gap, especially for case (c).
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/TNSE.2018.2813426, IEEE Transactions on Network Science and Engineering

\[
\sum_{i \neq j} |d_i - d_j| \times 10^4
\]

Fig. 8: Total degree difference over all nodes in the network for different link removal strategies. Lower values reflect the balance of degree distribution of the network.

\[
\sum_{i \neq j} |d_i - d_j|
\]

Fig. 9: Comparison of number of links involved in connectivity verification for different number of links removed at each iteration.

7 Conclusion

This paper proposes a distributed strategy for removing links from a network in the absence of global network structure information. The strategy is first to formulate a new optimization problem involving eigenvector corresponding to the largest eigenvalue and whose solution is equal to the original optimization problem. Distributed algorithm to estimate the eigenvector is then proposed which facilitates us to compute a suboptimal solution distributively. Even though in this paper we focus on link removal, the proposed distributed strategy is also applicable for computing suboptimal solution to link addition problem whose goal is to improve the network connectivity or maximize the spread of information in a network. Hence, the proposed distributed strategy has broad applications. Possible future work include solving link removal problem by considering the state of the nodes and also exploring alternative distributed strategy, e.g., by relaxing the binary decision variables (similar to the idea discussed in [45]) in combination with development of optimal threshold operation to recover the binary variables.

Acknowledgments

This work of A. Gusrialdi and Z. Qu is supported in part by US National Science Foundation under grants ECCS-1308928 and CCF-0956501, by US Department of Energy’s award DE-EE0006340, by US Department of Transportation’s award DTRT13-G-UTC51, by L-3 Communication’s contract 110132034, by Leidos’ contract P010161530, and by Texas Instruments’ award. The work of S. Hirche is funded by the German Research Foundation (DFG) within the Priority Program SPP 1914 on Cyber-Physical Networking.

References
