Distributed Algorithm for Link Removal in Directed Networks

Azwirman Gusrialdi

Tampere University, Pirkanmaa 33014, Finland
azwirman.gusrialdi@tuni.fi

Abstract. This paper considers the problem of removing a fraction of links from a strongly connected directed network such that the largest (in module) eigenvalue of the adjacency matrix corresponding to the network structure is minimized. Due to the complexity of the problem, an effective and scalable algorithm based on eigenvalue sensitivity analysis is proposed in the literature to compute the suboptimal solution to the problem. However, the algorithm requires knowledge of the global network structure and does not preserve strong connectivity of the resulting network. This paper proposes distributed algorithms which allow distributed implementation of the previously mentioned algorithm by relying solely on local information on the network topology while guaranteeing strong connectivity of the resulting network. A numerical example is provided to demonstrate the proposed distributed algorithm.

Keywords: link removal, strongly connected directed graph, distributed algorithm, optimization

1 Introduction

It is well-known that the dominant (largest in module) eigenvalue of the so-called adjacency matrix associated with a network plays an important role in the dissemination of an entity such as disease or information in both unidirectional and bidirectional networks. In other words, it determines whether the dissemination process will become an epidemic [1–5]. While there are several factors which affect dissemination process of an entity including the intrinsic property of the entity and the network topology, in this paper we assume that we could only modify the network structure where the entity spreads on. In particular, we focus on the problem of removing a fraction of links from a network in order to contain the dissemination by minimizing dominant eigenvalue of the network's adjacency matrix. The removal of links can be interpreted as controlling the interaction between people or cities in a country in order to slow the spread of disease when a vaccine is not yet available.

It is known that removing a fraction of links from a network to minimize the dominant eigenvalue of the adjacency matrix is a NP-hard problem [6]. In order to address this issue, several works have focused on developing strategies

to approximate and compute sub-optimal solution to this problem for both unidirectional and bidirectional networks, see for example [3,6–8]. An effective and scalable algorithm based on eigenvalue sensitivity analysis is presented in [3] to minimize dominant eigenvalue of the adjacency matrix by removing some links from a directed network. Specifically, an optimization problem involving the left and right eigenvectors corresponding the dominant eigenvalue is formulated to compute the sub-optimal solution. Note that the previously mentioned work assume that the global network structure is available and known to the designer. However, in practice the global network structure may not be available or may be very hard to obtain in a centralized manner due to geographical constraint or privacy concerns [9,10]. In addition to the availability of information on global network structure, the previously mentioned work do not take into account the (strong) connectivity of the network after the link removal. In some cases, it is desirable to preserve the (strong) connectivity of a network, for example so that important information can still be passed to all the users/nodes in the network or goods can still be transported between cities. Note that in [11], distributed algorithms which do not require knowledge of global network structure are proposed to remove a fraction of links from a network while guaranteeing the connectivity of the resulting network. However, the result is only limited to the case of bidirectional or undirected network.

The contribution of this paper is the development of distributed algorithms to compute the sub-optimal solution to link removal problem in a directed network while preserving strong connectivity of the resulting network. Specifically, matrix perturbation approach proposed in the literature is combined with novel distributed algorithms to estimate both the left and right dominant eigenvectors of the adjacency matrix to decide the candidate link to be removed. Furthermore, distributed verification algorithm is proposed to check whether a strongly connected directed network remains to be strongly connected after removing a fraction of links. This paper also generalizes the results presented in [11]. The proposed distributed algorithms can also readily be applied to the link addition problem whose goal is to maximize dominant eigenvalue of the adjacency matrix.

The organization of this paper is as follows: preliminaries followed by the problem formulation are presented in Section 2. The proposed distributed algorithms for link removal in directed networks are described in Section 3. A numerical example to demonstrate the proposed distributed strategy is provided in Section 4. Finally, Section 5 concludes the paper.

2 Problem Statement

In this section, we first provide a brief overview of graph theory and well-known results which will be used to develop distributed link removal strategy followed by the problem formulation.

2.1 Notation and Premilinaries

Let $\mathbb{R}$ be the set of real numbers and vector $\mathbf{1}_n \in \mathbb{R}^n$ 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 $\mathcal{V}$, $|\mathcal{V}|$ denotes the number of the elements in this set.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph (digraph) with a set of nodes $\mathcal{V} = \{1, 2, \cdots, n\}$ and a set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. An edge $(i, j) \in \mathcal{E}$ denotes that node i can obtain information from node j. The set of in-neighbors of node i is denoted by $\mathcal{N}^{\text{in}}_{\mathcal{G},i} = \{j | (i, j) \in \mathcal{E}\}$. Similarly, the set of out-neighbors of node i is denoted by $\mathcal{N}^{\text{out}}_{\mathcal{G},i} = \{j | (j, i) \in \mathcal{E}\}$. The directed graph $\mathcal{G}$ is *strongly connected* if every node can be reached from any other nodes by following a set of directed edges. For a matrix $C \in \mathbb{R}^{n \times n}$, let $[C]_{i*}$ and $[C]_{*i}$ represent vectors whose elements are equal to the i-th row and column of C respectively. Let us denote the dominant (i.e., largest in module) eigenvalue of matrix C as $\lambda(C)$. Matrix $C \in \mathbb{R}^{n \times n}$ is irreducible if and only if its associated graph $\mathcal{G}$ is strongly connected. The adjacency matrix associated with digraph $\mathcal{G}$, denoted by $A(\mathcal{G}) \in \mathbb{R}^{n \times n}$ is defined as

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

where $[A]_{ij}$ denote the element in the i-th row and j-th column of matrix A. Matrix C is nonnegative if all its elements are nonnegative. In addition, matrix C is primitive if it is irreducible and has at least one positive diagonal element [12].

Finally, we review a max-consensus algorithm which is one of the key elements in the proposed distributed link removal algorithm. Consider a strongly connected digraph $\mathcal{G}$ with n nodes and let us assign state $x_i(t) \in \mathbb{R}$ to each node of $\mathcal{G}$. If each node executes the following max-consensus algorithm [13]

$$x_i(t+1) = \max_{j \in \mathcal{N}^{\text{in}}_{\mathcal{G},i} \cup \{i\}} x_j(t) \tag{1}$$

all states $x_i(t)$ will then converge to $\max_i x_i(0)$ in no more than n steps.

2.2 Problem Formulation

Consider an n node network whose connection is given by a (unweighted) strongly connected directed graph $\mathcal{G}_0 = \{\mathcal{V}, \mathcal{E}_0\}$. From Perron-Frobenius theorem, it can be observed that $\lambda(A(\mathcal{G}_0))$ is real, strictly positive and simple [14]. For the sake of simplicity, we assume that the nodes know the network's size n. Otherwise, its value can be estimated distributively using the methods proposed in the literature, see for example [15]. Our objective is to remove at most m_e number of links $\Delta\mathcal{E}^-$ from $\mathcal{E}_0$ such that dominant eigenvalue of the adjacency matrix of resulting graph $\mathcal{G}_{m_e} = \{\mathcal{V}, \mathcal{E}_0 \setminus \Delta\mathcal{E}^-\}$ is minimized while also guaranteeing that $\mathcal{G}_{m_e}$ remains to be strongly connected. The problem can be formally formulated as the following optimization problem:

$$\begin{aligned} \min_{\Delta\mathcal{E}^-} \quad & \lambda(A(\mathcal{G}_{m_e})), \\ \text{s.t.} \quad & |\Delta\mathcal{E}^-| \leq m_e, \\ & \Delta\mathcal{E}^- \subseteq \mathcal{E}_0, \\ & \mathcal{G}_{m_e} \text{ is strongly connected.} \end{aligned} \tag{P1}$$

Solving optimization (P1) requires global knowledge on the network topology $\mathcal{G}_0$. However, in practice the global network topology $\mathcal{G}_0$ is often unknown or not available due to geographical constraint or privacy reasons such as in social network. Motivated by this issue, we impose the following constraint for the remaining of the paper.

Constraint 1 *The overall network topology $\mathcal{G}_0$ is not available. In addition, node i can only receive information via a communication network from nodes in the set $\mathcal{N}^{in}_{\mathcal{G}_0,i}$ and knows $\mathcal{N}^{out}_{\mathcal{G}_0,i}$.*

The absence of information on the overall network topology makes it impossible to solve (P1) in a centralized manner. Furthermore, optimization (P1) is a combinatorial problem whose complexity increases exponentially with the network size. Therefore, we are interested in developing a distributed strategy to compute the suboptimal solution to (P1) as stated in the following problem.

Problem 1. Assume that graph $\mathcal{G}_0$ is strongly connected. Find a suboptimal solution or an upper bound to the solution to optimization (P1) under constraint 1.

Remark 1. The local information available to each node as described in Constraint 1 is a standard assumption in distributed (cooperative) control literature, see for example references [16, 17].

3 Main Result

In order to solve Problem 1, we first adopt the strategy based on matrix perturbation theory presented in [3, 11]. Using matrix perturbation theory, for a graph with a large spectral gap (i.e., difference between the largest and second largest eigenvalue in magnitude) we can write

$$\lambda(A(\mathcal{G}_{m_e})) = \lambda(A(\mathcal{G}_0)) - \frac{\nu_0^T \Delta A^- w_0}{\nu_0^T w_0} + O(\|\Delta A^-\|^2) \tag{2}$$

where ΔA^- denotes the adjacency matrix corresponding to the graph whose links are given by $\Delta\mathcal{E}^-$. Moreover, ν_0, w_0 are the dominant left and right eigenvectors corresponding to $\lambda(A(\mathcal{G}_0))$, respectively. Due to the large spectral gap, we can neglect the higher order term in (2) and thus minimizing $\lambda(A(\mathcal{G}_{m_e}))$ is equivalent to maximizing $\nu_0^T \Delta A^- w_0/(\nu_0^T w_0)$. Defining the labeling $\ell_{\mathcal{G}_0} \in \{1, \cdots, |\mathcal{E}_0|\}$ on the edges of graph $\mathcal{G}_0$, matrix ΔA^- can be written as $\Delta A^- = \sum_{\ell_{\mathcal{G}_0}=1}^{|\mathcal{E}_0|} y_{\ell_{\mathcal{G}_0}} A_{\ell_{\mathcal{G}_0}}$ where $A_{\ell_{\mathcal{G}_0}}$ is a matrix with all zeros entries except for the ijth entry corresponding to the edge of label $\ell_{\mathcal{G}_0}$ which is equal to 1. Furthermore, $y_{\ell_{\mathcal{G}_0}} \in \{0, 1\}$ where $y_{\ell_{\mathcal{G}_0}} = 1$ means that the edge $\ell_{\mathcal{G}_0}$ in $\mathcal{E}_0$ is removed. Problem 1 can then be formulated as the following optimization problem

$$\begin{aligned} \max_{y} \quad & \frac{1}{\nu_0^T w_0} \sum_{\ell_{\mathcal{G}_0}=1}^{|\mathcal{E}_0|} y_{\ell_{\mathcal{G}_0}} \nu_{0,i} w_{0,j} \\ \text{s.t.} \quad & \mathcal{G}_{m_e} \text{ is strongly connected}, \\ & \mathbf{1}^T_{|\mathcal{E}_0|} y \leq m_e, \\ & y \in \{0,1\}^{|\mathcal{E}_0|} \end{aligned} \tag{P2}$$

where $\nu_{0,i}$ and $w_{0,i}$ respectively denotes the i-th element of left eigenvector ν_0 and w_0 associated with $\lambda(A(\mathcal{G}_0))$. In addition, the vector $y = [y_1, \cdots, y_{|\mathcal{E}_0|}]^T$. The analysis of optimality gap between the solutions obtained by solving (P2) and (P1) is discussed in [3]. In order to solve (P2), Note that ν_0, w_0 cannot be directly computed and whether graph $\mathcal{G}_{m_e}$ is strongly connected cannot be directly checked since the global network topology $\mathcal{G}_0$ is not available.

Next, we present distributed algorithms performed at each node, given that the nodes have local computational capability, to solve (P2) under constraint 1. To this end, we first define a primitive matrix Q_0 given by

$$Q_0 = I_n + A(\mathcal{G}_0) \tag{3}$$

where $I_n = \text{diag}(\mathbf{1}_n)$. Since matrix Q_0 is primitive, it is known that there exists a real dominant and simple eigenvalue of Q_0, denoted by $\lambda(Q_0)$ satisfying $\lambda(Q_0) > |\mu|$ for all other eigenvalues μ of Q_0 [14]. Hence, we have the following relationship: $\lambda(Q_0) = 1 + \lambda(A(\mathcal{G}_0))$. It can also be observed that both matrices Q_0 and $A(\mathcal{G}_0)$ share the same set of left and right eigenvectors (i.e., ν_0, w_0) which are both positive, up to rescaling [14].

3.1 Distributed Estimation of Dominant Right Eigenvector w_0

In this subsection we utilize power iteration method to estimate w_0 in a distributed manner. Specifically, each node performs the following iterations [18]:

$$\hat{w}_{0,i}(t+1) = \frac{1}{\|Q_0 \hat{w}_0(t)\|_\infty} \sum_{j \in \{\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i} \cup i\}} [Q_0]_{ij} \hat{w}_{0,j}(t) \tag{4}$$

where $\hat{w}_{0,i}(t)$ denotes the local estimation of $w_{0,i}$ at the t-th iteration. Note that since $w_0 > 0$, each node can choose any initial condition $\hat{w}_{0,i} > 0$. Furthermore, since the graph is strongly connected, it is guaranteed that under update law (4) local estimate $\hat{w}_{0,i}(t)$ will asymptotically converge to $w_{0,i}$ for all nodes i. Note that by using max-consensus algorithm (1) and by setting $x_i(0) = \sum_{j \in \{\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i} \cup i\}} [Q_0]_{ij} \hat{w}_{0,j}(t)$, each node will be able to compute $\|Q_0 \hat{w}_0(t)\|_\infty$ in a distributed manner. Therefore, update law (4) can then be implemented distributively by each node in the network. The nodes can implement the stopping criteria $\|\hat{w}_0(t) - \hat{w}_0(t-1)\|_\infty < \epsilon$ for a sufficiently small pre-defined threshold ϵ (to guarantee the estimation accuracy) which can also be checked in a distributed manner using max-consensus algorithm.

Remark 2. The normalization in (4) is performed to prevent the nonzero components in the iteration from becoming extremely large when $|\lambda| > 1$ or approaching zero if $|\lambda| < 1$. Hence, the normalization can be performed intermittently (which can be agreed by the nodes in advance before implementing the algorithm) since it has no effects on the convergence of power iteration method [19].

3.2 Distributed Estimation of Dominant Left Eigenvector ν_0

After estimating distributively the dominant right eigenvector w_0, the next step is to estimate the dominant left eigenvector ν_0 in a distributed manner. In contrast to the dominant right eigenvector, the distributed estimation of dominant left eigenvector has received less attention in the literature. To this end, we depart from the following relationship

$$Q_0^T \nu_0 = \lambda(Q_0)\nu_0, \tag{5}$$

where Q_0^T denotes the transpose of matrix Q_0. Each node can then distributively estimate ν_0 by solving (5) in a distributed fashion. First, observe that after estimating $w_{0,i}$ and from $Q_0 w_0 = \lambda(Q_0)w_0$, node i can estimate $\lambda(Q_0)$ according to

$$\lambda(Q_0) = \frac{[Q_0]_{i*}^T \hat{w}_0}{\hat{w}_{0,i}}. \tag{6}$$

Next, since node i knows $\mathcal{N}_{\mathcal{G}_0,i}^{out}$ it can construct the vector $[Q_0]_{*i}$ or $[Q_0^T]_{i*}$. In addition, after estimating $\lambda(Q_0)$ from (6), each node then estimates ν_0 by solving distributively a set of linear equations (5) which can be rewritten as

$$\underbrace{(Q_0^T - \lambda(Q_0)I_n)}_{\overline{Q}_0}\nu_0 = 0. \tag{7}$$

Specifically, the nodes cooperatively estimate ν_0 by performing the following iterations [20]:

$$\hat{\nu}_0^i(t+1) = \hat{\nu}_0^i(t) - P_i \left(\hat{\nu}_0^i(t) - \frac{1}{|\mathcal{N}_{\mathcal{G}_0,i}^{\text{in}}|} \sum_{j \in \mathcal{N}_{\mathcal{G}_0,i}^{\text{in}}} \hat{\nu}_0^j(t) \right) \tag{8}$$

ofwhere $\hat{\nu}_0^i(t)$ denotes the local estimation of ν_0 at node i at the t-th iteration and matrix P_i is defined as

$$P_i = I_n - [\overline{Q}_0]_{i*}([\overline{Q}_0]_{i*}^T[\overline{Q}_0]_{i*})^{-1}[\overline{Q}_0]_{i*}^T$$

which depends on local information of node i and $\overline{Q}_0$ is defined in (7). It should be noted that in general the set of linear equations (7) has many solutions. In order for local estimation $\hat{\nu}_0^i$ for $i = \{1, \cdots, n\}$ to converge to the same solution to (7), the initial condition of each node $\hat{\nu}_0^i(0)$ is chosen to minimize

$$\frac{1}{2}|\hat{\nu}_0^i(0) - b|^2 \text{ s.t. } [\overline{Q}_0]_{i*}^T \hat{\nu}_0^i(0) = 0 \tag{9}$$

for arbitrary vector $b > 0$ with $|\cdot|$ denotes the Euclidean norm. It is shown in [20] that under update law (8) whose initial conditions are chosen to minimize (9), all the nodes estimation $\hat{\nu}_0^i$ converge *exponentially fast* to the solution to (7) which is also the solution to: $\min_{\overline{Q}_0\nu_0=0} \frac{1}{2}|\nu_0 - b|^2$. The settling time of update law (8) can be calculated similar to the calculation in [21]. Note that update law (8) utilizes the same communication network $\mathcal{G}_0$ as the one utilized to distributively estimate w_0. Furthermore, in contrast to the estimation of w_0 presented in the previous subsection, node i will obtain the estimation of the full vector ν_0 instead of the i-th element $\nu_{0,i}$.

Remark 3. In comparison to distributed algorithm for estimating left and right eigenvectors corresponding to any irreducible matrices presented in [21] which requires each node to use memory $O(n^2)$ and to send n^2 values to its neighbors, the proposed distributed algorithm only requires to use memory $O(n)$ and to send n values to its neighbors. In addition, applying distributed estimation algorithms in [21] will reveal the global network structure to all nodes which may violate the privacy of each node. In contrast to [21], the proposed distributed algorithm respects the privacy in terms of the global network topology.

3.3 Distributed Verification of Digraph's Strong Connectivity

Let us assume that we remove a link $(i^*, j^*) \in \mathcal{E}_0$ from a strongly connected digraph $\mathcal{G}_0$. In this subsection we present a distributed algorithm based on max-consensus protocol to verify whether the resulting network $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$ remains to be strongly connected. To this end, each node is assigned a new variable $x_i(t) \in \mathbb{R}$ whose initial value is first set to $x_i(0) = 0, i = \{1, \cdots, n\}$. Given a candidate link to be removed (i^*, j^*), node j^* then modifies its initial value into $x_{j^*}(0) = 1$ while the remaining nodes do not change their initial values. All the nodes then execute max-consensus protocol (1) on the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$, that is node i^* does not use the information it received from node j^* (or node j^* does not send its information to node i^*) when executing the update law (1). We then have the following result on the relation between the final values of $x_i(t)$ and the strong connectivity of graph $\mathcal{G}_1$.

Lemma 1. *Given a strongly connected digraph $\mathcal{G}_0$ and a link $(i^*, j^*) \in \mathcal{E}_0$. Moreover, each node executes max-consensus protocol (1) on the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$ with initial values $x_{j^*}(0) = 1$ and $x_m(0) = 0$ for all $m \neq j^*$. The graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$ is strongly connected if and only if $x_i(n) = 1$ for all $i \in \mathcal{V}$.*

Proof. For showing the necessity ($\Longrightarrow$), since the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$ is strongly connected, it is shown in [13] that under max-consensus protocol (1) all nodes will converge to $\max_i x_i(0)$ which is equal to 1. Next, we show the sufficiency ($\Longleftarrow$). To do this note that the graph $\mathcal{G}_0$ is strongly connected. The removal of link (i^*, j^*) thus may result in that there exists no direct or indirect path from node j^* to node i^*. However, since we have $x_i(n) = 1$ under update law (1) for all nodes i in the network, this means that there is at least an indirect path from nodes j^* to i^*. Hence, the resulting graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$ remains to be strongly connected which completes the proof.

Remark 4. For the result in Lemma 1 to hold it requires the graph $\mathcal{G}_0$ to be strongly connected. In other words, the resulting graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (i^*, j^*)\}$ may not be necessarily strongly connected even though $x_i(n) = 1$ for all $i \in \mathcal{V}$ if the graph $\mathcal{G}_0$ is not strongly connected.

Remark 5. In contrast to the case of undirected network presented in [11], in the case of directed network the initial values in (1) cannot be chosen randomly

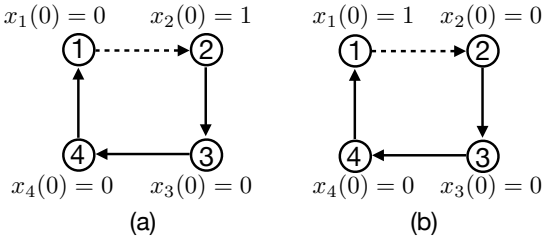

Fig. 1: Each node executes max-consensus protocol (1) on the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (2,1)\}$. (a) the graph $\mathcal{G}_1$ is not strongly connected even though $x_i(n) = 1$ for all nodes i when $x_2(0) = 1$ and $x_1(0) = 0$; (b) graph $\mathcal{G}_1$ is not strongly connected since $x_i(n) = 0$ for all nodes $i \neq 1$ when $x_1(0) = 1$ and $x_2(0) = 0$

between nodes i^* and j^* in order to check whether the resulting network is still strongly connected as illustrated in Fig. 1.

After each node executes update law (1) for n iterations with initial values described in Lemma 1, node i^* then checks whether $x_{i^*}(n) = 1$. If $x_{i^*}(n) = 1$, it needs to notify node j^* that the network remains to be strongly connected in the removal of link (i^*, j^*). To this end, each node is assigned with additional scalar variable $f_i(t)$. If the graph $\mathcal{G}_1$ is strongly connected (resp. not strongly connected), the initial values of f_i are set to $f_{i^*}(0) = 1$ (resp. $f_{i^*}(0) = -1$) and $f_m(0) = 0$ for all $m \neq i^*$. The nodes then again execute (1) on graph $\mathcal{G}_0$ with the previously described $f_i(0)$ and after n iterations, node j^* checks if $x_{j^*}(n) = 1$ (resp. $x_{j^*}(n) = 0$) then it will know that the the graph $\mathcal{G}_1$ remains to be strongly connected (resp. will not be strongly connected) after the removal of the link (i^*, j^*).

The strong connectivity of the resulting graph after removal of multiple links can then be distributively checked by iteratively applying the result in Lemma 1.

3.4 The Complete Distributed Link Removal Algorithm

After describing key elements required to develop the distributed algorithm, the pseudo-code of the complete distributed link removal algorithm to solve optimization problem (P2) is summarized in Algorithm 1.

Note that in steps 6 and 7 of Algorithm 1 it is assumed that the final estimation of left and right eigenvectors ν_0, w_0 have been normalized. For the left eigenvector ν_0, since each node has the estimation of the vector ν_0, i.e., $\hat{\nu}_0^i$, it can then normalize the estimation $\hat{\nu}_0^i$ independently. On the other hand, since node i only has the estimation of the i-th element of right eigenvector w_0, it then needs to normalize $\hat{w}_0$ cooperatively with the rest of the nodes. To this end, the norm $\|\hat{w}_0\|$ defined as $\|\hat{w}_0\| = \sqrt{\hat{w}_{0,1}^2 + \cdots + \hat{w}_{0,n}^2}$ can also be written as

$$\|\hat{w}_0\| = \sqrt{n\left(\frac{\hat{w}_{0,1}^2 + \cdots + \hat{w}_{0,n}^2}{n}\right)} = \sqrt{n\hat{w}_0^{ave}}.$$

Algorithm 1 Distributed algorithm to solve optimization problem (P2)

Require: $\mathcal{G}_0$ is strongly connected connected, node j can receive information from $\mathcal{N}^{\text{in}}_{\mathcal{G}_0,j}$ and knows $\mathcal{N}^{\text{out}}_{\mathcal{G}_0,j}$, n, m_e

$y = [0, \cdots, 0]^T \in \mathbb{R}^{|\mathcal{E}_0|}$
node j estimate $w_{0,j}$ using (4) whose estimation is given by $\hat{w}_{0,j}$
node j estimate the vector ν_p using (8) whose estimation is given by $\hat{\nu}^j_p$
initialize $p = 0$
while $p \leq m_e - 1$ **do**
  node j independently computes $(i^c, j) = \text{argmax}\, \hat{\nu}^j_{0,i} \hat{w}_{0,j}$ for $i \in \mathcal{N}^{\text{out}}_{\mathcal{G}_p,j}$
  all nodes compute $(i^*, j^*) = \text{argmax}\, \hat{\nu}^j_{0,i^c} \hat{w}_{0,j}$ with (i^c, j) obtained in the previous step using max-consensus (1) with $x_j(0) = \hat{\nu}^j_{0,i^c} \hat{w}_{0,j}$
  check strong connectivity of $\mathcal{G}_{p+1} = (\mathcal{V}, \mathcal{E}_p \setminus (i^*, j^*))$ using distributed algorithm described in Subsection 3.3
  if $\mathcal{G}_{p+1}$ is not strongly connected **then**
    back to steps 6–8 where node j^* excludes the link (i^*, j^*) when solving the optimization problem in step 6
    if $\mathcal{N}^{out}_{\mathcal{G}_p,i} = \emptyset$ for all i **then**
      break
    end if
  else
    continue to step 17
  end if
  $p \leftarrow p + 1$
  update $\mathcal{G}_p = \{\mathcal{V}, \mathcal{E}_{p-1} \setminus (i^*, j^*)\}$
  $y_{\ell^*_{\mathcal{G}_0}} = 1$ where $\ell^*_{\mathcal{G}_0} \sim (i^*, j^*)$
end while

If the nodes can compute $\hat{w}^{ave}_0$ distributively and given that they know n, each node can then compute $\|\hat{w}_0\|$. Specifically, the nodes can compute $\hat{w}^{ave}_0$ in a distributed manner using the finite-time average consensus algorithm proposed in the literature, e.g., 16, by setting its initial value as $\hat{w}^2_{0,i}$.

4 An Illustrative Example

In this section, we demonstrate the proposed distributed algorithm to compute solution to Problem 1. Consider a strongly connected digraph $\mathcal{G}_0$ consisting of eight nodes shown in Fig. 2 where each node may represent for example a city/state or a person. The number of links to be removed m_e is varied between 1 and 4. We choose a small size network so that the comparison with the centralized brute-force search approach, which in general is NP-hard, becomes possible. Interested reader is referred to the simulation results in 3 for the performance evaluation of solution to optimization problem (P2), without connectivity constraint, on real large graphs.

We apply Algorithm 1 to find solution to optimization problem (P2) for each m_e. As illustrated in Fig. 3a, the estimation $\hat{w}_{0,i}$ converges in less than 20 time steps to the true (unnormalized) right eigenvector $w_{0,i}$. Next, each node

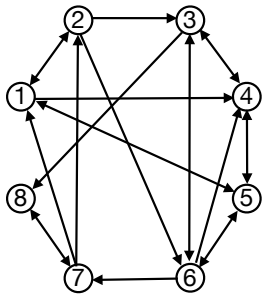

Fig. 2: A strongly connected directed graph used in the simulation

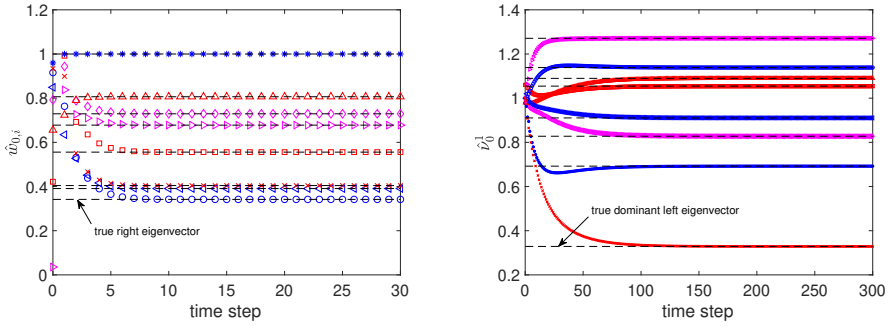

Fig. 3: (a) Estimated right eigenvector $\hat{w}_{0,i}$ corresponding to $\lambda(Q_0)$ with $Q_0 = A(\mathcal{G}_0) + I_n$ by each node (denoted by the markers) using power iteration method in (4). The estimation converge in less than 20 time steps; (b) Estimation of left eigenvector ν_0 corresponding to $\lambda(Q_0)$ by node 1 (i.e., $\hat{\nu}_0^1$)

distributively estimates the left eigenvector ν_0 using update law (8). Fig. 3b depicts the left eigenvector estimation of $A(\mathcal{G}_0)$ by node 1. As can be observed, the local estimation by each node converges to the true (unnormalized) left eigenvector ν_0. After estimating the eigenvectors (and normalizing them), the nodes then compute the candidate edge to be removed and check the strong connectivity of resulting graph. For comparison, we modify Algorithm 1 to iteratively and distributively remove one link at a time, that is after removing a link from the network we re-estimate the dominant left and right eigenvectors corresponding to the resulting network and compute the next link to be removed based on the new estimated dominant eigenvectors. In addition, to evaluate the optimality gap between the suboptimal and the global optimal solutions we also solve the original optimization problem (P1) by performing a brute-force search for each m_e and assuming that the global network topology is available.

The results are summarized in Table 1. First, it can be observed that for $m_e = 1$ the solutions to (P1) and (P2) are the same, i.e., the optimality gap is equal to zero. For the case of $m_e = 2, 3, 4$ there is a gap between the val-

Table 1: Comparison of solution using different strategies

m_e	Algorithm 1 Optimization (P2)		Iterative link removal Optimization (P2)		Brute-force search Optimization (P1)	
	$\Delta\mathcal{E}^-$	$\lambda(A(\mathcal{G}_{m_e}))$	$\Delta\mathcal{E}^-$	$\lambda(A(\mathcal{G}_{m_e}))$	$\Delta\mathcal{E}^-$	$\lambda(A(\mathcal{G}_{m_e}))$
1	(5,4)	2.5209	(5,4)	2.5209	(5,4)	2.5209
2	(5,4), (6,5)	2.3717	(5,4), (6,3)	2.2426	(5,4), (6,3)	2.2426
3	(5,4), (6,5) (6,3)	2.0826	(5,4), (6,3) (2,1)	2.0295	(2,1), (6,3) (6,5)	2.0135
4	(5,4), (6,5) (6,3), (1,5)	1.9728	(5,4), (6,3) (2,1), (6,5)	1.8216	(5,1), (6,3) (5,4), (2,7)	1.8111

ues of $\lambda(A(\mathcal{G}_{m_e}))$ corresponding to the solution obtained from Algorithm 1 and by applying brute-force search. However, the gap could be made smaller if we iteratively remove one link at a time for each m_e. In fact, when $m_e = 2$ the optimality gap between iterative link removal and brute force search is equal to zero, i.e., there is no performance loss in spite of the absence of the global network topology. Intuitively, one of the reasons is because when we remove iteratively one link at a time, the matrix perturbation ΔA^- in (2) is sufficiently small so that the term $\nu_p^T \Delta A^- w_p/(\nu_p^T w_p)$ at iteration p could predict the movement of eigenvalue $\lambda(A(\mathcal{G}_p))$ when it is perturbed by ΔA^-.

5 Conclusion

This paper proposes eigenvalue sensitivity based distributed algorithm to remove a fraction of links from a strongly connected directed network such that dominant eigenvalue of the adjacency matrix is minimized. In addition, the algorithm also guarantees that the resulting network remains to be strongly connected after the link removals. The proposed distributed algorithms consist of distributed estimation of both left and right eigenvectors corresponding to the largest (in module) eigenvalue of the adjacency matrix together with distributed verification algorithm to check whether a network is strongly connected after removal of a link. A numerical example demonstrates the implementation and efficacy of the proposed distributed algorithm.

Acknowledgement This work is supported by Academy of Finland under academy project decision number 330073.

References

1. Wang, Y., Chakrabarti, D., Wang, C., Faloutsos, C.: Epidemic spreading in real networks: An eigenvalue viewpoint. In: Proc. 22nd International Symposium on Reliable Distributed Systems, pp. 25–34 (2003)
2. Prakash, B.A., Chakrabarti, D., Valler, N., Faloutsos, M., Faloutsos, C.: Threshold conditions for arbitrary cascade models on arbitrary networks. Knowledge and Information Systems **33**(3), 549–575 (2012)

3. Chen, C., Tong, H., Prakash, B.A., Eliassi-Rad, T., Faloutsos, M., Faloutsos, C.: Eigen-optimization on large graphs by edge manipulation. ACM Transactions on Knowledge Discovery from Data (TKDD) **10**(4), 49 (2016)
4. Li, C., Wang, H., Van Mieghem, P.: Epidemic threshold in directed networks. Physical Review E **88**(6), 062,802 (2013)
5. Van Mieghem, P., Van de Bovenkamp, R.: Non-markovian infection spread dramatically alters the susceptible-infected-susceptible epidemic threshold in networks. Physical review letters **110**(10), 108,701 (2013)
6. Van Mieghem, P., Stevanović, D., Kuipers, F., Li, C., van de Bovenkamp, R., Liu, D., Wang, H.: Decreasing the spectral radius of a graph by link removals. Physical Review E **84**, 016,101 (2011)
7. Bishop, A.N., Shames, I.: Link operations for slowing the spread of disease in complex networks. Europhysics Letters **95**(18005)
8. Milanese, A., Sun, J., Nishikawa, T.: Approximating spectral impact of structural perturbations in large networks. Physical Review E **81**(4), 046,112 (2010)
9. McDaniel, P., McLaughlin, S.: Security and privacy challenges in the smart grid. Security Privacy, IEEE **7**(3), 75–77 (2009)
10. Li, N., Zhang, N., Das, S.: Preserving relation privacy in online social network data. IEEE Internet Computing **15**(3), 35–42 (2011)
11. Gusrialdi, A., Qu, Z., Hirche, S.: Distributed link removal using local estimation of network topology. IEEE Transactions on Network Science and Engineering **6**(3), 280–292 (2019)
12. Horn, R., Johnson, C.R.: Matrix Analysis. Cambridge University Press, New York (2009)
13. Nejad, B., Attia, S., Raisch, J.: Max-consensus in a max-plus algebraic setting: The case of fixed communication topologies. In: International Symposium on Information, Communication and Automation Technologies, pp. 1–7 (2009)
14. Bullo, F.: Lectures on Network Systems, 1 edn. CreateSpace (2018). URL http://motion.me.ucsb.edu/book-lns With contributions by J. Cortes, F. Dorfler, and S. Martinez
15. Shames, I., Charalambous, T., Hadjicostis, C.N., Johansson, M.: Distributed network size estimation and average degree estimation and control in networks isomorphic to directed graphs. In: Proc. of Annual Allerton Conference on Communication, Control, and Computing, pp. 1885–1892. IEEE (2012)
16. Charalambous, T., Yuan, Y., Yang, T., Pan, W., Hadjicostis, C.N., Johansson, M.: Distributed finite-time average consensus in digraphs in the presence of time delays. IEEE Transactions on Control of Network Systems **2**(4), 370–381 (2015)
17. Cai, K., Ishii, H.: Average consensus on general strongly connected digraphs. Automatica **48**(11), 2750–2761 (2012)
18. Golub, G.H., Van Loan, C.F.: Matrix Computations (3rd Ed.). Johns Hopkins University Press, Baltimore, MD, USA (1996)
19. Ghaboussi, J., Wu, X.S.: Numerical Methods in Computational Mechanics. CRC Press (2016)
20. Wang, X., Mou, S., Sun, D.: Improvement of a distributed algorithm for solving linear equations. IEEE Transactions on Industrial Electronics **64**(4), 3113–3117 (2016)
21. Gusrialdi, A., Qu, Z.: Distributed estimation of all the eigenvalues and eigenvectors of matrices associated with strongly connected digraphs. IEEE Control Systems Letters **1**(2), 328–333 (2017)