Network Science (2022), 1–19
doi:10.1017/nws.2022.25

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Connectivity-preserving distributed algorithms for removing links in directed networks

Azwirman Gusrialdi

Faculty of Engineering and Natural Sciences, Tampere University, Pirkanmaa, Finland
Email: azwirman.gusrialdi@tuni.fi

Action Editor: Christoph Stadtfeld

Abstract

This article considers the link removal problem in a strongly connected directed network with the goal of minimizing the dominant eigenvalue of the network's adjacency matrix while maintaining its strong connectivity. Due to the complexity of the problem, this article focuses on computing a suboptimal solution. Furthermore, it is assumed that the knowledge of the overall network topology is not available. This calls for distributed algorithms which rely solely on the local information available to each individual node and information exchange between each node and its neighbors. Two different strategies based on matrix perturbation analysis are presented, namely simultaneous and iterative link removal strategies. Key ingredients in implementing both strategies include novel distributed algorithms for estimating the dominant eigenvectors of an adjacency matrix and for verifying strong connectivity of a directed network under link removal. It is shown via numerical simulations on different type of networks that in general the iterative link removal strategy yields a better suboptimal solution. However, it comes at a price of higher communication cost in comparison to the simultaneous link removal strategy.

Keywords: link removal; strongly connected digraph; distributed algorithm; estimation; optimization; information exchange; maximum consensus

1. Introduction

1.1 Motivation and problem description

Dominant (largest in module) eigenvalue of the 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 particular, the dominant eigenvalue determines whether a dissemination process will become an epidemic (Wang et al., 2003; Prakash et al., 2012; Chen et al., 2016; Li et al., 2013; Van Mieghem & Van de Bovenkamp, 2013). Dissemination process of an entity can be affected by several factors including the intrinsic property of the entity and the network topology. In this article, it is assumed that one could modify only the network structure where the entity spreads on. Specifically, this article focuses on the problem of removing a fraction of links/edges from a network with the goal of containing the dissemination by minimizing the dominant eigenvalue of the network's adjacency matrix. This problem 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. In addition, in practice it is also desirable to preserve the (strong) connectivity of a network. For example, strong connectivity of a network ensures that important information can still be passed to all the users/nodes in the network or goods can still be delivered between the cities.

The problem of removing a fraction of links from a network to minimize the dominant eigenvalue of the adjacency matrix poses several challenges. First, the problem is NP-hard due to its combinatorial nature (Van Mieghem et al., 2011). Second, 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 (McDaniel & McLaughlin, 2009; Li et al., 2011). The absence of the overall network topology makes the design of an efficient algorithm for removing a fraction of links very challenging. This article aims at developing link removal algorithms by addressing simultaneously both challenges described previously.

1.2 Related literature

Since the link removal problem is NP-hard, most of the work focuses on heuristics and approximations of the problem. That is, they focused on developing strategies to approximate and compute a suboptimal solution to this problem for both unidirectional and bidirectional networks, see for example (Bishop & Shames, 2011; Van Mieghem et al., 2011; Chen et al., 2016; Milanese et al., 2010; Yang et al., 2016; Wu et al., 2017). Among these work, an effective and scalable algorithm based on eigenvalue sensitivity analysis is proposed in (Chen et al., 2016) to minimize the dominant eigenvalue of an adjacency matrix by removing a fraction of links from a directed network. To this end, a suboptimal solution is computed by solving an optimization problem which involves both the left and right eigenvectors associated with the dominant eigenvalue of the adjacency matrix. However, all the previously mentioned work including (Chen et al., 2016) suffer from the following limitations: (1) the global network structure is assumed to be available and known to the designer; (2) the strong connectivity of the resulting network is not guaranteed. It is worth to note that the authors in (Gusrialdi et al., 2019) proposed distributed algorithms based on eigenvalue sensitivity analysis which do not require knowledge of the overall network structure to remove a fraction of links from a network. Moreover, the distributed strategy also ensures the connectivity of the resulting network. However, the application of the proposed strategy is only limited to bidirectional or undirected network and its extension to directed network is highly nontrivial.

1.3 Statement of contribution

The contribution of this article is the development of distributed algorithms to compute a suboptimal solution to the link removal problem in a directed network while preserving strong connectivity of the resulting network. Specifically, with the help of matrix perturbation theory, the problem of minimizing dominant eigenvalue of an adjacency matrix is reformulated as an optimization problem involving both the left and right eigenvectors corresponding to the dominant eigenvalue of the adjacency matrix. Novel distributed algorithms to estimate both the dominant left and right eigenvectors are then presented which will be used to decide the candidate links to be removed. In addition, distributed algorithm is proposed to verify whether the removal of a fraction of links will destroy or maintain strong connectivity of the resulting directed network. This article is an extensively extended version of the conference paper (Gusrialdi, 2021). Specifically, in this article: (1) new distributed algorithms, including the one with reduced communication cost, for estimating the dominant eigenvectors are proposed; (2) a formal proof of the proposed distributed link removal algorithms is presented; (3) a new iterative link removal strategy is presented in details and its performance comparison with the simultaneous link removal strategy is evaluated on different type of networks. It is worth to note that even though in this article we focus on link removal problem, the proposed distributed algorithms can be readily applied to the link addition problem whose goal is to maximize dominant eigenvalue of the network's adjacency matrix.

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

1.4 Organization of the article

This article is organized as follows: preliminaries followed by the problem formulation are presented in Section 2. The proposed distributed link removal algorithms are presented in Section 3. The proposed distributed algorithms are demonstrated and evaluated using several numerical simulations in Section 4. Finally, Section 5 concludes the article.

2. Problem statement

In this section, we provide a brief overview of graph theory and maximum/minimum consensus algorithms followed by the problem formulation.

2.1 Notation and preliminaries

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. The number of the elements in the set $\mathscr{V}$ is denoted by $|\mathscr{V}|$. The graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ is a directed graph (digraph) with a set of nodes $\mathscr{V} = \{1, 2, \cdots, n\}$ and a set of edges (links) $\mathscr{E} \subset \mathscr{V} \times \mathscr{V}$. Let $(j, i) \in \mathscr{E}$ represents a directed edge from node j to node i. When the graph $\mathscr{G}$ represents a communication network topology, the edge $(j, i) \in \mathscr{E}$ denotes that node j can send information to node i or node i can receive information from node j. The set of in-neighbors of node i is denoted by $\mathscr{N}^{\text{in}}_{\mathscr{G},i} = \{j | (j, i) \in \mathscr{E}\}$. Similarly, the set of out-neighbors of node i is denoted by $\mathscr{N}^{\text{out}}_{\mathscr{G},i} = \{j | (i, j) \in \mathscr{E}\}$. The directed graph $\mathscr{G}$ 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 us denote its dominant (i.e., largest in module) eigenvalue as $\lambda(C)$. The adjacency matrix associated with digraph $\mathscr{G}$, denoted by $A(\mathscr{G}) \in \mathbb{R}^{n\times n}$ is defined as

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

where $[A]_{ij}$ denotes the element in the ith row and jth column of matrix A. Matrix $C \in \mathbb{R}^{n\times n}$ is nonnegative (i.e., $C \geq 0$) if all its elements are nonnegative. A nonnegative matrix C is irreducible if and only if $(I_n + C)^{n-1} > 0$ where I_n denotes an identity matrix of size n. Matrix C is primitive if it is irreducible and has at least one positive diagonal element.

Finally, we review the maximum and minimum consensus algorithms which will be used in development of the distributed link removal algorithms. Consider a digraph $\mathscr{G}$ with n nodes and assume that each node maintains a state or variable $x_i(t) \in \mathbb{R}$ at discrete time $t = \{0, 1, 2, \cdots\}$. Let T denote the maximum of the shortest path length between any pair of nodes in $\mathscr{G}$. Furthermore, each node executes the following maximum consensus algorithm

$$x_i(t + 1) = \max_{j\in\mathscr{N}^{\text{in}}_{\mathscr{G},i}\cup\{i\}} x_j(t). \tag{2}$$

It is shown in Nejad et al. (2009) that $x_i(t) = x_j(t) = \max_{k\in\mathscr{V}} x_k(0)$ for all $i, j \in \mathscr{V}$, $\forall t \geq T$, and any initial conditions $x_i(0)$ *if and only if* the digraph $\mathscr{G}$ is strongly connected. Analogously, for the minimum consensus algorithm, each node executes

$$x_i(t + 1) = \min_{j\in\mathscr{N}^{\text{in}}_{\mathscr{G},i}\cup\{i\}} x_j(t) \tag{3}$$

and as a result the states of all nodes converge to the value $\min_{i\in\mathscr{V}} x_i(0)$ after T steps if and only if the digraph $\mathscr{G}$ is strongly connected.

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

2.2 Problem formulation

Consider an n node network whose connection is given by an unweighted strongly connected directed graph $\mathcal{G}_0 = \{\mathcal{V}, \mathcal{E}_0\}$. It is known that the dominant eigenvalue $\lambda(A(\mathcal{G}_0))$ is real, strictly positive, and simple (Bullo, 2018). Our objective is to remove at most m_e number of links $\Delta\mathcal{E}^-$ from the set $\mathcal{E}_0$ 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 guaranteeing $\mathcal{G}_{m_e}$ remains to be strongly connected. The problem can be formally formulated as the following optimization problem:

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

In general, the global knowledge on the network topology $\mathcal{G}_0$ is required to solve the optimization (P1). However, the global network topology $\mathcal{G}_0$ is often unknown or not available in practice due to geographical constraint or privacy reasons. Motivated by this limitation, the following constraint is imposed for the remaining of the article.

Constraint 1. *The overall network topology $\mathcal{G}_0$ is not available. Node i can receive information via a communication network only from nodes in the set $\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}$. Furthermore, node i knows the set $\mathcal{N}^{\text{out}}_{\mathcal{G}_0,i}$. In other words, node i only knows the ith row and column of matrix A.*

The absence of information on the overall network topology prevents us from solving (P1) in a centralized manner. In addition, optimization (P1) is a combinatorial problem whose complexity increases exponentially with the network size. Therefore, in this article, we are interested in developing a distributed strategy to compute a 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.*

To this end, we assume that each individual node is equipped with both computational and data storage capabilities in addition to communication via a network whose structure is similar to $\mathcal{G}_0$. Furthermore, for the sake of simplicity, it is assumed that the nodes know the network's size n. Alternatively, the network's size can be estimated distributively using the methods proposed in the literature, see for example (Shames et al., 2012).

It is worth to note that one possible strategy to overcome Constraint 1 is by passing local information of each node (e.g., $\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}$) to the rest of the network so that in the end each node can construct the global network topology $\mathcal{G}_0$. However, this flooding strategy is not locally adaptable to the changes in the network topology. More importantly, this strategy will also reveal the global network topology to all the nodes in the network, which is not desirable due to privacy concerns.

3. Main results: proposed distributed link removal algorithms

A suboptimal solution to optimization problem (P1) can be computed using the matrix perturbation theory presented in e.g., (Chen et al., 2016; Gusrialdi et al., 2019). Specifically, for a graph with a large spectral gap (i.e., difference between the largest and second largest eigenvalue in magnitude), the dominant eigenvalue $\lambda\big(A(\mathcal{G}_{m_e})\big)$ can be written as

$$\lambda\big(A(\mathcal{G}_{m_e})\big) = \lambda(A(\mathcal{G}_0)) - \frac{v_0^T \Delta A^- w_0}{v_0^T w_0} + O\big(\|\Delta A^-\|^2\big) \tag{4}$$

where ΔA^- denotes the adjacency matrix corresponding to the graph whose links are given by the set $\Delta\mathcal{E}^-$. Furthermore, v_0, w_0 denote the dominant left and right eigenvectors corresponding to

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

eigenvalue $\lambda(A(\mathscr{G}_0))$, respectively. Since the spectral gap of the graph $\mathscr{G}_0$ is large, the higher order term in Equation (4) can be neglected and thus minimizing $\lambda\left(A\left(\mathscr{G}_{m_e}\right)\right)$ is equivalent to maximizing the term $v_0^T \Delta A^- w_0 / \left(v_0^T w_0\right)$. Therefore, a suboptimal solution to optimization problem (P1) is given by the set of edges $\Delta\mathscr{E}^-$ which solves the following optimization problem

$$\begin{aligned} \max_{\Delta\mathscr{E}^- \subseteq \mathscr{E}_0} \quad & \frac{1}{v_0^T w_0} \sum_{(j,i)\in\Delta\mathscr{E}^-} v_{0,i} w_{0,j} \\ \text{s.t.} \quad & |\Delta\mathscr{E}^-| \leq m_e, \\ & \mathscr{G}_{m_e} \text{ is strongly connected,} \end{aligned} \tag{P2}$$

where $v_{0,i}$ and $w_{0,i}$, respectively, denote the ith element of left eigenvector v_0 and w_0 associated with $\lambda(A(\mathscr{G}_0))$. Analysis of the optimality gap between the solutions obtained by solving optimization problems (P2) and (P1) are discussed in Chen et al. (2016).

Optimization problem (P2) involves the dominant left and right eigenvectors associated with the graph $\mathscr{G}_0$. However, since the global network topology $\mathscr{G}_0$ is not available, the dominant eigenvectors v_0, w_0 cannot be directly computed and similarly, strong connectivity of the resulting directed graph $\mathscr{G}_{m_e}$ also cannot be directly verified. Therefore, as a first step in solving (P2), it is necessary to develop distributed algorithms performed at each node to estimate the dominant eigenvectors v_0, w_0 under Constraint 1. To this end, let us define the primitive matrix Q_0 as

$$Q_0 = cI_n + A(\mathscr{G}_0), \;\; c \in \mathbb{R}^+. \tag{5}$$

For simplicity, in the remaining of the article we choose $c = 1$. 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 the other eigenvalues μ of Q_0 (Bullo, 2018). Hence, we have the following relationship:

$$\lambda(Q_0) = 1 + \lambda(A(\mathscr{G}_0)). \tag{6}$$

Furthermore, it can also be seen that both matrices Q_0 and $A(\mathscr{G}_0)$ share the same set of left and right eigenvectors (i.e., v_0, w_0) which are both positive, up to rescaling (Bullo, 2018). In order to distributively estimate the dominant eigenvectors v_0, w_0, we will work with matrix Q_0 instead of $A(\mathscr{G}_0)$. The proposed strategy can be summarized as follows:

1. estimate in a distributed manner the dominant eigenvalue $\lambda(Q_0)$
2. using the estimated dominant eigenvalue $\lambda(Q_0)$, the dominant right and left eigenvectors are then computed by solving distributively the following linear equations

$$\begin{aligned} (Q_0 - \lambda(Q_0) I_n) w_0 = 0, \\ \left(Q_0^T - \lambda(Q_0) I_n\right) v_0 = 0. \end{aligned} \tag{7}$$

Details of the proposed strategy are described below.

3.1 Distributed algorithm for estimating dominant eigenvalue

In order to distributively estimate the dominant eigenvectors, each node is first required to estimate the dominant eigenvalue $\lambda(Q_0)$. To this end, let us assign to each node a scalar variable $h_i \in \mathbb{R}$ with initial value $h_i(0) > 0$. Each node then updates their variables h_i according to the following rule

$$h_i(t+1) = \sum_{j \in \mathscr{N}^{in}_{\mathscr{G}_0,i} \cup i} [Q_0]_{ij} h_j(t). \tag{8}$$

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

In addition, let us define

$$\underline{\lambda}(t) = \min_{1\le i\le n} \frac{h_i(t+1)}{h_i(t)}, \qquad \overline{\lambda}(t) = \max_{1\le i\le n} \frac{h_i(t+1)}{h_i(t)}. \tag{9}$$

We then have the following relations between scalars $\underline{\lambda}(t), \overline{\lambda}(t)$, and dominant eigenvalue $\lambda(Q_0)$.

$$\underline{\lambda}(0) \le \underline{\lambda}(1) \le \ldots \le \lambda(Q_0) \le \ldots \le \overline{\lambda}(1) \le \overline{\lambda}(0). \tag{10}$$

Since matrix Q_0 is non-negative and primitive, it is known that $\underline{\lambda}(t)$ and $\overline{\lambda}(t)$ will converge to $\lambda(Q_0)$ (Wood & O'Neill, 2004).

It can be observed that the update law (8) can be performed at each node in a distributed manner using only local information available to individual node. Furthermore, the scalar variables $\underline{\lambda}(t)$ (resp. $\overline{\lambda}(t)$) can also be computed in a distributed manner using minimum (resp. maximum) consensus algorithm (3) (resp. (2)). For example, in order to compute $\overline{\lambda}(1)$ distributively, each node maintains a variable $x_i(t)$ which is the local estimate of $\overline{\lambda}(1)$ and sets its initial value as $x_i(0) = \frac{h_i(2)}{h_i(1)}$. The local estimate $x_i(t)$ is then updated according to Equation (2) and after n steps, all the local estimates are equal to $\overline{\lambda}(1)$.

It is worth to note that the update rule given in Equation (8) is a power method and it is shown that $h_i(t)$ will converge to the right dominant eigenvector w_0 from which the dominant eigenvalue can be computed (Golub & Van Loan, 1996). However, instead of estimating the dominant right eigenvector directly using the power method, in this article we first estimate the dominant eigenvalue since, as can be observed from Equation (10), both the estimates $\underline{\lambda}(t)$ and $\overline{\lambda}(t)$ provide a bound on the estimation error of $\lambda(Q_0)$. By taking advantage of this error bound, the designer can then provide a desired accuracy which will be used as a stopping criteria for the estimation of the dominant eigenvalue $\lambda(Q_0)$. That is, for a sufficiently small threshold $\varepsilon > 0$, the nodes computes (8), (9) until

$$|\overline{\lambda}(t) - \underline{\lambda}(t)| < \varepsilon. \tag{11}$$

3.2 Distributed algorithm for estimating dominant left and right eigenvectors

Given the estimated dominant eigenvalue $\lambda(Q_0)$, the next step is to distributively estimate the right and left dominant eigenvectors by solving linear equations (7) in a distributed manner. Before proceeding, let us define the following matrices:

$$\overline{Q} = Q_0 - \lambda(Q_0) I_n,$$
$$\widetilde{Q} = Q_0^T - \lambda(Q_0) I_n.$$

Hence, linear equations in (7) can be written as

$$\overline{Q} w_0 = 0, \tag{12a}$$

$$\widetilde{Q} v_0 = 0. \tag{12b}$$

The matrices $\overline{Q}$ and $\widetilde{Q}$ can be partitioned as

$$\overline{Q} = \begin{bmatrix} \overline{Q}_1 \\ \overline{Q}_2 \\ \vdots \\ \overline{Q}_n \end{bmatrix}, \quad \widetilde{Q} = \begin{bmatrix} \widetilde{Q}_1 \\ \widetilde{Q}_2 \\ \vdots \\ \widetilde{Q}_n \end{bmatrix} \tag{13}$$

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

where $\overline{Q}_i \in \mathbb{R}^{1\times n}, \widetilde{Q}_i \in \mathbb{R}^{1\times n}$ for $i \in \{1, 2, \cdots, n\}$. Since the dominant eigenvalue $\lambda(Q_0)$ has been estimated and from the Constraint 1, the ith node knows both vectors $\overline{Q}_i$ and $\widetilde{Q}_i$.

First, let us look at the distributed estimation of the right dominant eigenvector w_0 using only local information available to each node. To this end, each node maintains a variable $\hat{w}_0^i(t) \in \mathbb{R}^n$ which is a local estimate of w_0 at node i. The local estimate of each node is then updated according to the following rule by exchanging information with its in-neighbors set $\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}$:

$$\hat{w}_0^i(t+1) = \hat{w}_0^i(t) - \overline{P}_i \left(\hat{w}_0^i(t) - \frac{1}{|\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}|} \sum_{j\in\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}} \hat{w}_0^j(t) \right) \tag{14}$$

where matrix $\overline{P}_i$ is defined as

$$\overline{P}_i = I_n - \overline{Q}_i^T (\overline{Q}_i \overline{Q}_i^T)^{-1} \overline{Q}_i$$

and the initial condition $\hat{w}_0^i(0)$ is chosen to satisfy $\overline{Q}_i \hat{w}_0^i(0) = 0$ and $\hat{w}_0^i(0) \neq 0$. Under update rule (14), all the local estimates $\hat{w}_0^i$ will converge exponentially fast to the same solution to $\overline{Q} w_0 = 0$, i.e., the right dominant eigenvector w_0 as shown in Mou et al. (2015).

Similarly, in order to estimate the dominant left eigenvector by solving distributively linear equations (12b), each node maintains a local estimate $\hat{v}_0^i(t) \in \mathbb{R}^n$ updated according to the following rule

$$\hat{v}_0^i(t+1) = \hat{v}_0^i(t) - \widetilde{P}_i \left(\hat{v}_0^i(t) - \frac{1}{|\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}|} \sum_{j\in\mathcal{N}^{\text{in}}_{\mathcal{G}_0,i}} \hat{v}_0^j(t) \right) \tag{15}$$

where matrix $\widetilde{P}_i$ is defined as

$$\widetilde{P}_i = I_n - \widetilde{Q}_i^T (\widetilde{Q}_i \widetilde{Q}_i^T)^{-1} \widetilde{Q}_i$$

and the initial condition $\hat{v}_0^i(0)$ is chosen to satisfy $\widetilde{Q}_i \hat{w}_0^i(0) = 0$ and $\hat{v}_0^i(0) \neq 0$. Note that both distributed algorithms (14), (15) can be performed using the same strongly connected communication network topology $\mathcal{G}_0$.

In contrast to distributed estimation algorithms presented in (Gusrialdi & Qu, 2017) which requires each node to send n^2 number of values to its neighbors, using distributed algorithms (14), (15) each node only needs to send n number of values, that is the local estimate $\hat{w}_0(t)$ or $\hat{v}_0(t)$ to its neighbors. However, when the network's size n is large, it either will be communication-costly for individual node to send the entire estimate vector $\hat{w}_0^i(t)$ or $\hat{v}_0^i(t)$ to its neighbors at each time or it is not possible to do so if the node has limited communication capacity. To address this issue, we next present an alternative distributed algorithm based on the method proposed in (Liu & Anderson, 2020) to solve linear equations (12b), (12a) with a reduced communication cost.

First, the local estimate $\hat{w}_0^i(t)$ and $\hat{v}_0^i(t)$ are partitioned as

$$\hat{w}_0^i(t) = \begin{bmatrix} y_{i1}(t) \\ y_{i2}(t) \\ \vdots \\ y_{i\ell}(t) \end{bmatrix}, \quad \hat{v}_0^i(t) = \begin{bmatrix} z_{i1}(t) \\ z_{i2}(t) \\ \vdots \\ z_{i\ell}(t) \end{bmatrix}, \tag{16}$$

where $\ell > 1$ is a positive integer and $y_{i\sigma}(t), z_{i\sigma}(t)$ with $\sigma \in \{1, 2, \cdots, \ell\}$ are vectors. Note that the local estimates can be partitioned in an arbitrary manner, that is ℓ can be chosen to be any

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

positive integer and $y_{i\sigma}(t), z_{i\sigma}(t)$ can have arbitrary size depending on the node's communication. Furthermore, each discrete-time time $t \geq 1$ can be uniquely written in the following form

$$t = k\ell + \sigma(t), \;\; k \in \{0, 1, 2, \cdots\}, \;\; \sigma(t) \in \{1, 2, \cdots, \ell\}.$$

Next, let us define

$$\hat{w}_0^{i\sigma(t)}(t) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ y_{i\sigma(t)}(t) \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \;\; \hat{v}_0^{i\sigma(t)}(t) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ z_{i\sigma(t)}(t) \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

At each time $t \geq 1$, each node broadcasts the vectors $y_{i\sigma(t)}(t)$ and $z_{i\sigma(t)}(t)$ to its neighbors and updates its local estimates according to

$$\hat{w}_0^i(t+1) = \hat{w}_0^i(t) - \overline{P}_i \left(\hat{w}_0^{i\sigma(t)}(t) - \frac{1}{|\mathcal{N}_{\mathcal{G}_0,i}^{\text{in}}|} \sum_{j \in \mathcal{N}_{\mathcal{G}_0,i}^{\text{in}}} \hat{w}_0^{j\sigma(t)}(t) \right) \tag{17}$$

and

$$\hat{v}_0^i(t+1) = \hat{v}_0^i(t) - \widetilde{P}_i \left(\hat{v}_0^{i\sigma(t)}(t) - \frac{1}{|\mathcal{N}_{\mathcal{G}_0,i}^{\text{in}}|} \sum_{j \in \mathcal{N}_{\mathcal{G}_0,i}^{\text{in}}} \hat{v}_0^{j\sigma(t)}(t) \right). \tag{18}$$

It is shown in (Liu & Anderson, 2020) that under update law (17) (resp. (18)), the local estimate $\hat{w}_0^i(t)$ (resp. $\hat{v}_0^i(t)$) for all nodes converge exponentially fast to the same solution to linear equations (12a) (resp. (12b)), that is the dominant eigenvector w_0 (resp. v_0).

Intuitively, distributed algorithms (14), (15) converge faster to the solution to linear equations (12a) and (12b) compared to distributed algorithms (17), (18) since the full entries of vectors $\hat{w}_0^i(t)$ and $\hat{v}_0^i(t)$ are being exchanged in Equations (14), (15) at each time t. Therefore, the designer could choose the size of vectors $y_{i\sigma}(t), z_{i\sigma}(t)$ and integer $\ell > 1$ by taking into account the trade-off between the communication capacity and convergence speed. In addition, under distributed algorithms (17) and (18), each node can exchange the vectors $y_{i\sigma(t)}(t)$ and $z_{i\sigma(t)}(t)$ at the same time so that the dominant right and left eigenvectors can be estimated simultaneously.

3.3 Distributed algorithm for verifying digraph's strong connectivity

After developing distributed algorithms for estimating both the right and left dominant eigenvectors, the next step is to develop distributed algorithm to verify the strong connectivity of a digraph when a link $(j^*, i^*) \in \mathcal{E}_0$ is being removed. In other words, given a candidate link to be removed $(j^*, i^*) \in \mathcal{E}_0$, we want to verify in a distributed manner whether the resulting network $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (j^*, i^*)\}$ remains to be strongly connected.

To this end, all the nodes execute maximum consensus protocol (2) on the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \backslash (j^*, i^*)\}$, i.e., node j^* does not send its information to node i^* when executing the update law (2). The initial values of $x_i(t)$ are chosen as

$$x_{j^*}(0) = 1 \;\; \text{and} \;\; x_m(0) = 0 \text{ for all nodes } m \neq j^*. \tag{19}$$

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

Algorithm 1. Distributed connectivity verification for $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_0 \setminus (j^*, i^*))$

Require: $\mathcal{G}_0 = (\mathcal{V}, \mathcal{E}_0)$ is strongly connected and a link to be removed $(j^*, i^*) \in \mathcal{E}_0$

set the initial values of protocol (2) as in Equation (19)
execute maximum consensus (2) for n iterations on graph $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_0 \setminus (j^*, i^*))$
if $x_i(n) = 1$ for all $i \in \mathcal{V}$ **then**
    the graph $\mathcal{G}_1$ is strongly connected
else
    $\mathcal{G}_1$ is not strongly connected
end if
execute again maximum consensus (2) for n iterations on graph $\mathcal{G}_0$ with initial values in Equation (20)
if $x_{j^*}(n) = 1$ **then**
    node j^* knows that the graph $\mathcal{G}_1$ is strongly connected
else
    node j^* knows that the graph $\mathcal{G}_1$ is not strongly connected
end if

The following result reveals the relationship between the final values of $x_i(t)$ and the strong connectivity of graph $\mathcal{G}_1$.

Proposition 1. *Given a strongly connected digraph $\mathcal{G}_0$ and a link $(j^*, i^*) \in \mathcal{E}_0$. Each node executes max-consensus protocol (2) on the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \setminus (j^*, i^*)\}$ with initial values $x_{j^*}(0) = 1$ and $x_m(0) = 0$ for all $m \neq j^*$. The graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \setminus (j^*, i^*)\}$ is strongly connected if and only if $x_i(n) = 1$ for all $i \in \mathcal{V}$.*

Proof. For showing the necessity ($\Longrightarrow$), observe that since the graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \setminus (j^*, i^*)\}$ is strongly connected, that max-consensus protocol (2) makes all the states $x_i(t)$ converge (after n steps) to $\max_i x_i(0)$ which is equal to 1. In order to show the sufficiency ($\Longleftarrow$), note that the removal of link (j^*, i^*) 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 (2) for all nodes i in the network, this means that there exists at least one indirect path from node j^* to i^*. Hence, it can be concluded that the resulting graph $\mathcal{G}_1 = \{\mathcal{V}, \mathcal{E}_0 \setminus (j^*, i^*)\}$ remains to be strongly connected. □

After each node executes update law (2) for n iterations with initial values described in Proposition 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 after removing link (j^*, i^*). This can be done by again executing the maximum consensus algorithm (2) on graph $\mathcal{G}_0$ whose initial values are chosen as

$$x_{i^*}(0) = \begin{cases} 1 & \text{if } \mathcal{G}_1 \text{ is strongly connected} \\ -1 & \text{if otherwise} \end{cases}, \quad x_m(0) = 0 \text{ for all } m \neq i^*. \tag{20}$$

If $x_{j^*}(n) = 1$ (resp. $x_{j^*}(n) = 0$) then node j^* will know that the graph $\mathcal{G}_1$ remains to be strongly connected (resp. will not be strongly connected) when the link (j^*, i^*) is being removed. Algorithm 1 summarizes distributed verification of the link removal. Note that the strong connectivity of the resulting graph under multiple links removal can be verified by applying iteratively Algorithm 1 for each individual candidate link to be removed.

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

Algorithm 2. Distributed algorithm for finding m_e links with largest value of $v_{0,i}w_{0,j}$ whose removal preserves graph's strong connectivity

Require: $\mathcal{G}_0$ is strongly connected connected, number of nodes n, number of links to be removed m_e, estimated dominant eigenvectors w_0, v_0 using distributed algorithms (14) (or (17)) and (15) (or (18)), Each edge is label by $\ell_{j,i}$ in a descending order according to the value $v_{0,i}w_{0,j}$ using similar steps as below and by ignoring the steps 6–14.

1: normalize the estimated dominant eigenvectors w_0, v_0
2: initialize $p = 0$
3: **while** $p \leq m_e - 1$ **do**
4: node j independently computes $(j, i^c) = \text{argmax } \hat{v}^j_{0,i}\hat{w}^j_{0,j}$ for $i \in \mathcal{N}^{\text{out}}_{\mathcal{G}_p,j}$
5: all nodes compute $(j^*, i^*) = \text{argmax } \hat{v}^j_{0,i^c}\hat{w}^j_{0,j}$ with (j, i^c) obtained in the previous step using max-consensus (2) with $x_j(0) = \hat{v}^j_{0,i^c}\hat{w}^j_{0,j}$
6: check strong connectivity of $\mathcal{G}_{p+1} = (\mathcal{V}, \mathcal{E}_p \setminus (j^*, i^*))$ using Algorithm 1
7: **if** $\mathcal{G}_{p+1}$ is not strongly connected **then**
8: back to steps 4–6 where node j^* excludes the link (j^*, i^*) which destroys strong connectivity of the resulting graph
9: **if** $\mathcal{N}^{out}_{\mathcal{G}_p,i} = \emptyset$ for all i **then**
10: **break**
11: **end if**
12: **else**
13: continue to step 18
14: **end if**
15: $p \leftarrow p + 1$
16: update $\mathcal{G}_p = \{\mathcal{V}, \mathcal{E}_{p-1} \setminus (j^*, i^*)\}$
17: $\mathcal{I}_1 \leftarrow \ell_{(j^*,i^*)}$
18: **end while**
19: Set of links with m_e largest value of $v_{0,i}w_{0,j}$ which preserves connectivity is $\mathcal{I}_1$

3.4 The complete distributed algorithms for link removal

In this subsection, we present distributed algorithms for solving optimization problem (P2) using the ingredients developed in the previous subsections. Before proceeding, let us introduce the following notation. For a given set $\mathcal{I}_i$, its smallest and largest elements, denoted, respectively, by $\underline{k}_i$ and $\overline{k}_i$, are defined as

$$\underline{k}^i = \min_{k \in \mathcal{I}_i} k, \quad \overline{k}^i = \max_{k \in \mathcal{I}_i} k. \tag{21}$$

Furthermore, let $\ell_{(j,i)} \in \{1, 2, \cdots, |\mathcal{E}_0|\}$ be the label of the edges (j, i) of graph $\mathcal{G}_0$ in a descending order according to the value $v_{0,i}w_{0,j}$ of the edges. That is, if $v_{0,i}w_{0,j} > v_{0,p}w_{0,q}$ then $\ell_{(j,i)} < \ell_{(q,p)}$ and if two edges have the same value of $v_{0,i}w_{0,j}$, e.g., $v_{0,i}w_{0,j} = v_{0,p}w_{0,q}$, then the label can be assigned as $\ell_{(j,i)} = \ell_{(q,p)} + 1$ or $\ell_{(q,p)} = \ell_{(j,i)} + 1$. Hence, an edge with label $\ell_{(j,i)}$ also means that the edge (j, i) has the $\ell_{(j,i)}$th largest value of $v_{0,i}w_{0,j}$. First, it can be observed that the solution to optimization problem (P2) in the absence of connectivity preserving constraint is given by a set of links with m_e largest value of $v_{0,i}w_{0,j}$ labeled by $\ell_{(j,i)} = 1, 2, \cdots, m_e$. The solution can be easily obtained using Algorithm 2 and by ignoring steps 6–14 in it. However, the introduction of additional constraint

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

Algorithm 3. Distributed algorithm for solving optimization problem (P2)

Require: $\mathscr{G}_0$ is strongly connected connected, number of nodes n, number of links to be removed m_e, estimated dominant eigenvectors w_0, v_0 using distributed algorithms (14) (or (17)) and (15) (or (18))

compute the set $\mathscr{I}_1$ using Algorithm 2
compute the link label $\ell^*_{(j,i)}$ from Equation (22)
initialize $d = 2$
while $d \leq \bar{\ell}^1_{(j,i)}$ **do**
  set $v_{0,i}w_{0,j} = 0$ for the links labeled by $\underline{\ell}^s_{(j,i)}$ with $s = 1, \cdots, d-1$
  find set of links with m_e largest value of $v_{0,i}w_{0,j}$ which preserves graph's connectivity using similar method to Algorithm 2. Store the solution in $\mathscr{I}_d$
  if $\underline{\ell}^d_{(j,i)} > \ell^*_{(j,i)}$ **then**
    break
  end if
  $d \leftarrow d + 1$
end while
The solution to optimization (P2), that is $\Delta\mathscr{E}^-$ equals to $\mathscr{I}_{i^*} = \text{argmax}_{\mathscr{I}_s} \sum_{(j,i)\in\mathscr{I}_s} v_{0,i}w_{0,j}$

to preserve strong connectivity of the network makes solving optimization problem (P2) more challenging.

In the following we describe in a less-formal way the idea of the proposed distributed algorithms for solving optimization problem (P2). Details of the algorithms are summarized in Algorithm 3. Broadly speaking, the algorithms aim at finding link candidate sets $\mathscr{I}_i$ for $i = 1, 2, \cdots$, where $|\mathscr{I}_i| = m_e$ and one of the sets $\mathscr{I}_i$ is the solution to (P2). This strategy is similar to a brute-force search, but instead of looking at all possible link combinations we only look at a small number of combinations. Specifically, the distributed algorithms consist of the following main steps.

1. Find links given by the set $\mathscr{I}_1$ with m_e largest value of $v_{0,i}w_{0,j}$ whose removal do not destroy strong connectivity of the network
2. Compute the edge's label $\ell^*_{(j,i)}$ which is the solution to the following optimization

$$\begin{aligned} \min_{\ell_{(j,i)}\in\mathscr{I}_1} \quad & \ell_{(j,i)} \\ \text{s.t.} \quad & \left|\ell_{(j,i)} - \bar{\ell}^1_{(j,i)}\right| \leq m_e, \end{aligned} \tag{22}$$

   where the labels $\bar{\ell}^1_{(j,i)}$ is defined according to (21).
3. After finding the first link candidate set $\mathscr{I}_1$, the other link candidate sets $\mathscr{I}_k$ with $k = 2, 3, \cdots$ are computed by repeating step 1 as long as the resulting $\underline{\ell}^k_{(j,i)}$ of the set $\mathscr{I}_k$, defined according to (21), satisfies $\underline{\ell}^k_{(j,i)} \leq \ell^*_{(j,i)}$ and for each $k = 2, 3, \cdots$ we set the value of $v_{0,i}w_{0,j}$ equal to zero for the links labeled by $\underline{\ell}^s_{(j,i)}$ with $s = 1, \cdots, k-1$. This step in principle provides information on how far we should explore the possible candidates of the optimal links.
4. The solution to optimization (P2) is then given by the set $\mathscr{I}_{i^*}$ whose links have the largest value of $\sum_{(j,i)\in\mathscr{I}_k} v_{0,i}w_{0,j}$

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

The following theorem shows that the set of links computed by Algorithm 3 is the solution to optimization problem (P2).

Theorem 1. *A set of links obtained from Algorithm 3 is the solution to optimization problem (P2).*

Proof. Assume that the link candidate sets computed from Algorithm 3 is given by $\mathscr{I}_k$ with $k = 1, 2, \cdots, \alpha$. The set of links, denoted by $\mathscr{I}_{\alpha+1}$ with m_e largest value of $v_{0,i}w_{0,j}$ whose smallest label equals to $\ell^*_{(j,i)} + 1$ is given by

$$\mathscr{I}_{\alpha+1} = \left\{ \ell^*_{(j,i)} + 1, \ell^*_{(j,i)} + 2 + \cdots, \ell^*_{(j,i)} + m_e \right\}. \tag{23}$$

Hence, it is sufficient to prove that

$$\sum_{(j,i)\in\mathscr{I}_1} v_{0,i}w_{0,j} \geq \sum_{(j,i)\in\mathscr{I}_{\alpha+1}} v_{0,i}w_{0,j} \tag{24}$$

for the set $\mathscr{I}_1$ with smallest value of $\sum_{(j,i)\in\mathscr{I}_1} v_{0,i}w_{0,j}$. This is because

$$\sum_{(j,i)\in\mathscr{I}_{i^*}} v_{0,i}w_{0,j} \geq \sum_{(j,i)\in\mathscr{I}_1} v_{0,i}w_{0,j}$$

for the optimal solution set of links $\mathscr{I}_{i^*}$ as can be seen in step 12 of Algorithm 3. To this end, for a given label $\bar{\ell}^1_{(j,i)}$ the set $\mathscr{I}_1$ with possible smallest value $\left(\text{i.e., when } \left|\ell^*_{(j,i)} - \bar{\ell}^1_{(j,i)}\right| = m_e \text{ and } \underline{\ell}^1_{(j,i)} = \ell^*_{(j,i)}\right)$ of $\sum_{(j,i)\in\mathscr{I}_1} v_{0,i}w_{0,j}$ is given by

$$\mathscr{I}_1 = \left\{ \ell^*_{(j,i)}, \ell^*_{(j,i)} + 2, \ell^*_{(j,i)} + 3, \cdots, \ell^*_{(j,i)} + m_e - 1, \ell^*_{(j,i)} + m_e \right\}. \tag{25}$$

Recalling that an edge with label $\ell_{(j,i)}$ also means that the edge (j, i) has the $\ell_{(j,i)}$th largest value of $v_{0,i}w_{0,j}$, it can be seen that (24) is satisfied for the sets given in (23) and (25). This completes the proof. □

4. Numerical simulations

In this section, the proposed distributed algorithms are demonstrated and evaluated using several simulations.

4.1 Comparison with the optimal solution

First, we are interested in evaluating the optimality gap between the proposed distributed algorithms and the brute-force search. To this end, we consider a strongly connected graph consisting of 10 nodes as shown in Figure 1. Choosing a small size network allows us to compare the solutions obtained by the proposed distributed algorithms with the ones obtained from the brute-force search (given that the global network topology is available) which is in general NP-hard. Interested reader is referred to (Chen et al., 2016) for the performance evaluation of Algorithm 3 on real large graphs without connectivity constraint.

The number of links to be removed m_e in the numerical simulation is varied between 1 and 6. We then apply Algorithm 3 to solve optimization problem (P2). First, each node distributively estimates the dominant eigenvalue of matrix Q_0 in Equation (5) based on update rule (8), (9) whose estimation is depicted in Figure 2.

It can be observed that the estimation converges to the dominant eigenvalue within few time steps. Using the estimated dominant eigenvalue, the nodes then distributively estimate both the dominant right and left eigenvectors using update rule (14) (or (17) for reduced communication

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

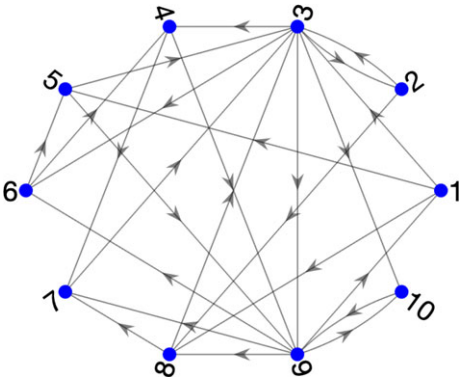

Figure 1. Distributed estimation of dominant eigenvalue $\lambda(Q_0)$ using (8), (9).

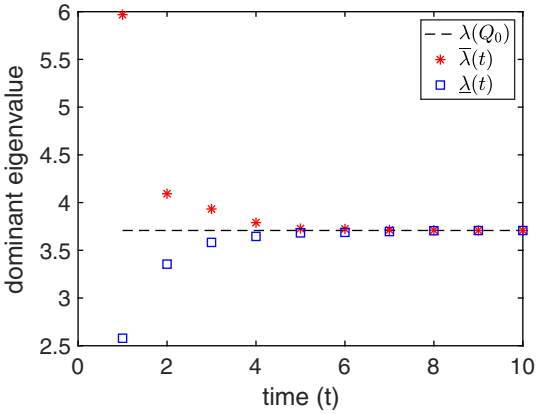

Figure 2. Distributed estimation of dominant eigenvalue $\lambda(Q_0)$ using (8), (9).

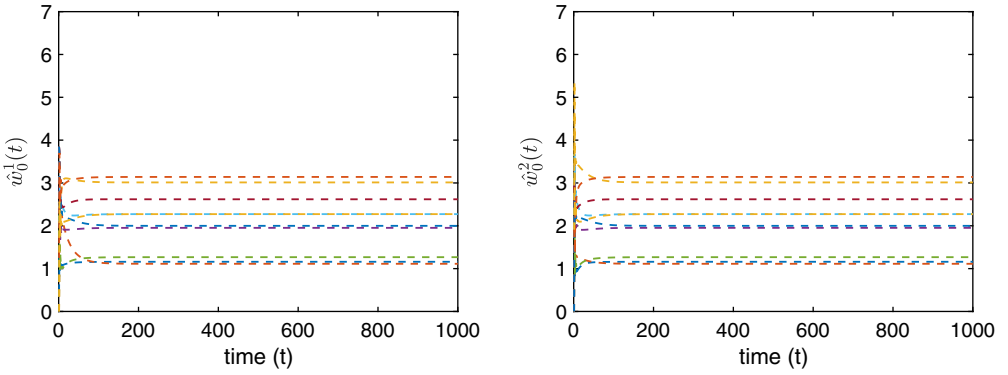

Figure 3. Distributed estimation of dominant right eigenvector by node 1 (left) and node 2 (right) using update law (14).

cost with $\ell = 5$) and (15) (or (18) for reduced communication cost with $\ell = 5$), respectively. Estimation of the dominant eigenvectors by nodes 1 and 2 are illustrated in Figures 3–6. It can be observed that the nodes could successfully estimate both dominant eigenvectors. Furthermore, the estimation based on update rule (14) and (15) converge faster in comparison to the ones based

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

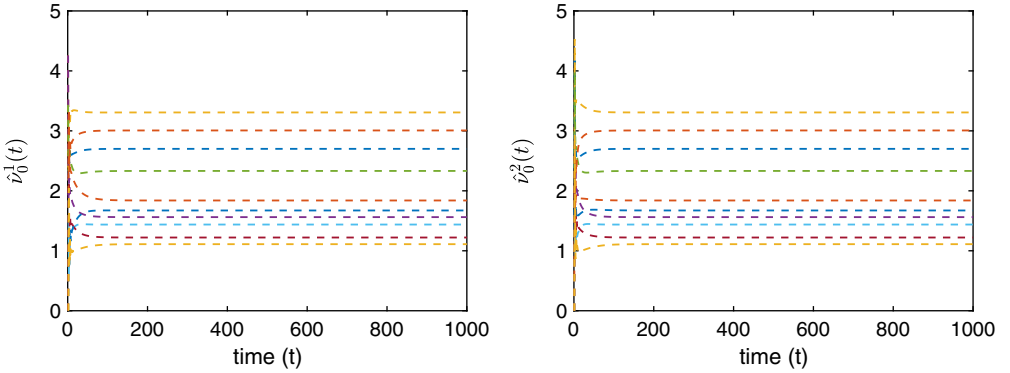

Figure 4. Distributed estimation of dominant left eigenvector by node 1 (left) and node 2 (right) using update law (15).

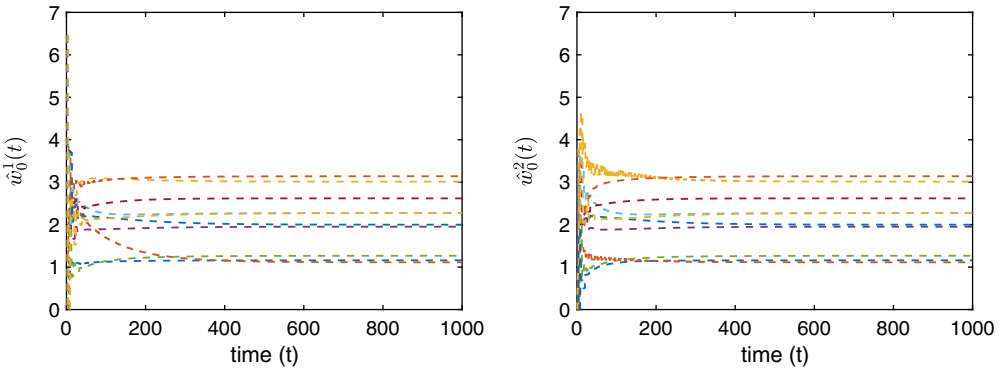

Figure 5. Distributed estimation of dominant right eigenvector with reduced communication cost by node 1 (left) and node 2 (right) using update law (17).

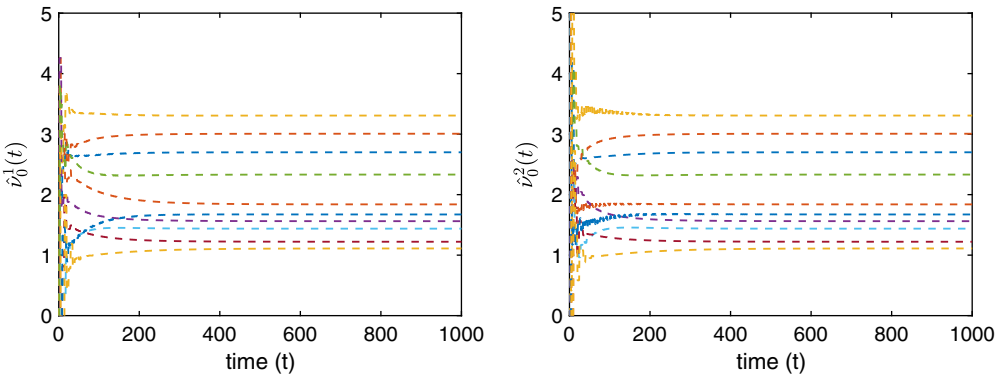

Figure 6. Distributed estimation of dominant left eigenvector with reduced communication cost by node 1 (left) and node 2 (right) using update law (18).

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

Table 1. Comparison of solutions using different strategies

	Simultaneous removal Algorithm 3		Iterative removal Algorithm 4		Brute-force search Optimization (P1)	
m_e	$\Delta\mathscr{E}^-$	$\lambda(A(\mathscr{G}_{m_e}))$	$\Delta\mathscr{E}^-$	$\lambda(A(\mathscr{G}_{m_e}))$	$\Delta\mathscr{E}^-$	$\lambda(A(\mathscr{G}_{m_e}))$
1	(3,9)	2.4715	(3,9)	2.4715	(3,9)	2.4715
2	(3,9), (8,3)	2.3404	(3,9), (4,9)	2.2520	(3,9), (4,9)	2.2520
3	(3,9), (8,3), (4,9)	2.1320	(3,9), (4,9), (6,5)	2.0582	(3,9), (3,10), (4,9)	2.0414
4	(3,9), (8,3), (4,9), (6,5)	1.9239	(3,9), (4,9), (6,5), (8,3)	1.9239	(3,4), (3,9), (3,10), (6,5)	1.9194
5	(3,9), (8,3), (4,9), (6,5), (9,8)	1.8939	(3,9), (4,9), (6,5), (8,3), (2,3)	1.8142	(2,3), (3,9), (4,9), (6,5), (8,3)	1.8142
6	(3,9), (8,3), (4,9), (6,5), (9,8), (3,4)	1.8291	(3,9), (4,9), (6,5), (8,3), (2,3), (9,10)	1.7090	(2,3), (3,9), (4,9), (6,5), (8,3), (9,10)	1.7090

on update rule (17) and (18) where only two elements of the estimated eigenvectors are being exchanged with the other nodes at each time step.

After distributively estimating the dominant eigenvectors, the nodes then solve optimization problem (P2) using Algorithm 3. The solutions to optimization (P2) obtained from Algorithm 3 and the ones obtained by solving optimization (P1) using the brute-force search (assuming the knowledge of overall network topology) for different values of m_e are summarized in Table 1. As can be observed, when $m_e = 1$ both Algorithm 3 and brute-force search result in the same solution, i.e., zero optimality gap. However, as m_e increases the gap between the solutions obtained from Algorithm 3 and brute-force search becomes larger. It is demonstrated in (Gusrialdi et al., 2019) for the case of undirected network that iteratively solving optimization (P2), i.e., removing one link at a time followed by re-estimating the dominant eigenvectors of the resulting network to compute the next link to be removed, may result in a better performance in comparison to removing m_e simultaneously as done in Algorithm 3. Motivated by this observation, next we apply the iterative link removal strategy to solve optimization (P2) whose details are given in Algorithm 4. The results are summarized in Table 1. It can be observed that for $m_e = 1, 2, 5, 6$ the iterative link removal strategy result in the same solution to the ones obtained from the brute-force search, i.e., the optimality gap is zero. In addition, for $m_e = 3, 4$ the gap with the optimal solution is smaller for the iterative link removal strategy (i.e., Algorithm 4) compared to the simultaneous link removal strategy (i.e., Algorithm 3). It should be noted that in contrast to simultaneous link removal, the iterative link removal strategy requires a longer time and higher communication cost for computing the solutions as the dominant eigenvectors need to be re-estimated after removal of each individual link from the network.

4.2 Comparison between simultaneous and iterative link removal strategies

Next, we compare the performance of both simultaneous and iterative link removal strategies for different types of network. To this end, we consider four types of networks illustrated in Figure 7, namely: (a) modular small-world network which is a random, directed network with a specified number of fully connected modules linked together by randomly distributed between-module connections; (b) random network with specified number of nodes and edges; (c) ring lattice network (with toroidal boundary conditions); and (d) non-ring lattice network (without toroidal boundary conditions). For the simulations, the number of links to be removed is varied between 1 and 20% of the total number of links.

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

Algorithm 4. Distributed algorithm for iterative link removal

Require: $\mathcal{G}_0$ is strongly connected connected, number of nodes n, number of links to be removed m_e

initialize $p = 0$
while $p \leq m_e - 1$ **do**
  each node estimates and normalizes w_p using (14) or (17) whose estimation is given by $\hat{w}_p^i$
  each node estimates and normalizes v_p using (15) or (18) whose estimation is given by $\hat{v}_p^i$
  node j independently computes $(j, i^c) = \text{argmax } \hat{v}_{p,i}^j \hat{w}_{p,j}^j$ for $i \in \mathcal{N}_{\mathcal{G}_p,j}^{\text{out}}$
  all nodes compute $(j^*, i^*) = \text{argmax } \hat{v}_{p,i^c}^j \hat{w}_{p,j}^j$ with (j, i^c) obtained in the previous step using max-consensus (2) with $x_j(0) = \hat{v}_{p,i^c}^j \hat{w}_{p,j}^j$
  check strong connectivity of $\mathcal{G}_{p+1} = (\mathcal{V}, \mathcal{E}_p \setminus (j^*, i^*))$ using Algorithm 1
  if $\mathcal{G}_{p+1}$ is not strongly connected **then**
    back to steps 5–7 where node j^* excludes the link (j^*, i^*) which destroys strong connectivity of the resulting graph
    if $\mathcal{N}_{\mathcal{G}_p,i}^{out} = \emptyset$ for all i **then**
      break
    end if
  else
    continue to step 16
  end if
  $p \leftarrow p + 1$
  update $\mathcal{G}_p = \{\mathcal{V}, \mathcal{E}_{p-1} \setminus (j^*, i^*)\}$
  $\Delta\mathcal{E}^- \leftarrow (j^*, i^*)$
end while

The results of applying Algorithm 3 and Algorithm 4 to the networks described previously are summarized in Figures 8–11. From the results, we can make the following observations:

- For the cases of random, ring lattice and nonring lattice networks, the iterative link removal strategy yields better performance (i.e., lower value of $\lambda(A(\mathcal{G}_{m_e}))$ means better performance) compared to simultaneous link removal strategy. Furthermore, the performance gap between both strategies tend to increase as the value of m_e increases. Note that for random and ring-lattice networks, both strategies performed similarly when the number of links to be removed is less than 5% as can be seen in Figure 9 and Figure 10.
- For nonring lattice network, the simultaneous link removal strategy could not reduce the dominant eigenvalue that much even though around 20% of total links have been removed from the network, see Figure 11. Hence, for this type of network, it is recommended to use iterative link removal strategy which comes at a price of higher communication cost.
- For the modular small-world network, the performance of both algorithms depend on the cluster size for a fixed number of nodes and edges. As can be observed in Figure 8, when the cluster size is small the iterative link removal strategy performed better compared to simultaneous link removal strategy. However, as the cluster size increases, the simultaneous link removal strategy outperformed the iterative link removal strategy for some values of m_e.

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

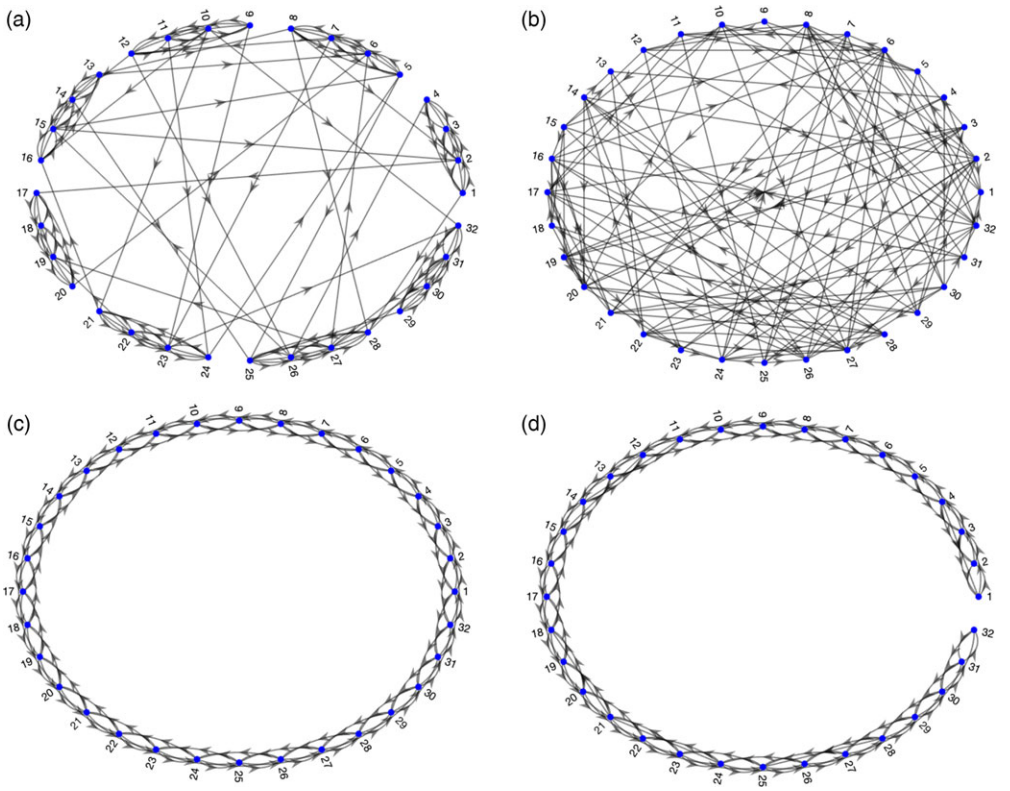

Figure 7. Examples of networks used in the simulation: (a) modular small-world network; (b) random network; (c) ring lattice network; (d) nonring lattice network.

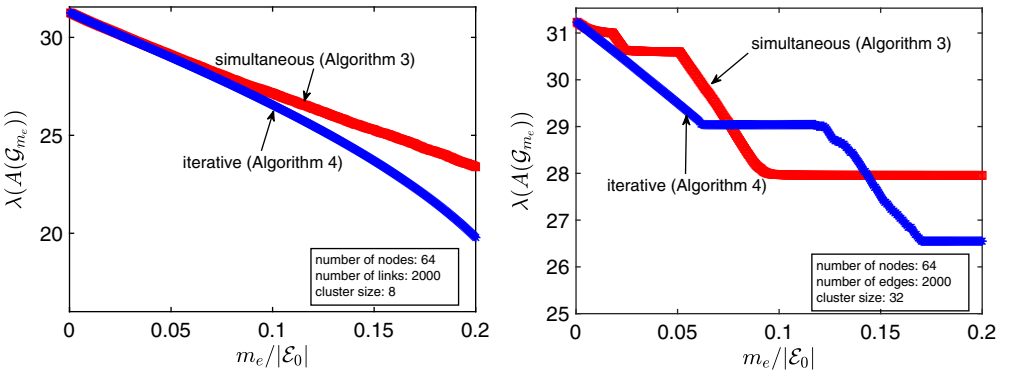

Figure 8. Comparison between simultaneous and iterative link removal strategies on modular small-world network.

5. Conclusions

This article proposed two novel distributed algorithms, namely simultaneous and iterative link removal strategies to compute a suboptimal solution to link removal problem while maintaining network's strong connectivity whose objective is to minimize the dominant eigenvalue of the adjacency matrix associated with the network's structure. Key ingredients include distributed algorithms for estimating the dominant eigenvectors of an adjacency matrix together with distributed algorithm to verify network's strong connectivity after the removal of a fraction of links. It was shown in simulations on different type of networks that there was a trade-off between the

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

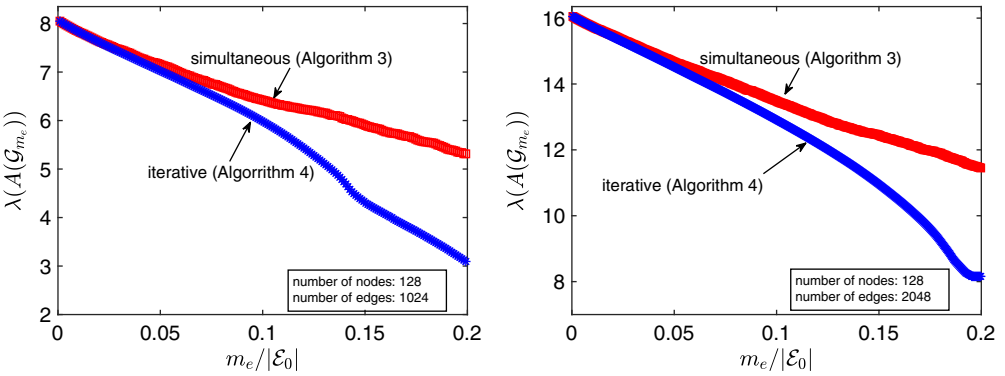

Figure 9. Comparison between simultaneous and iterative link removal strategies on random network.

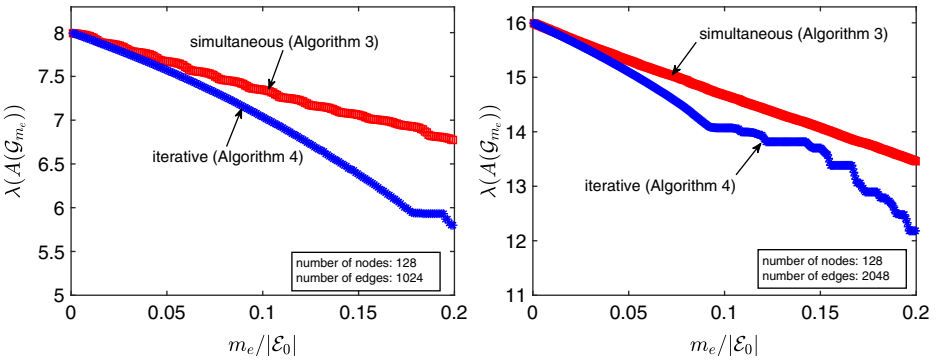

Figure 10. Comparison between simultaneous and iterative link removal strategies on ring lattice network.

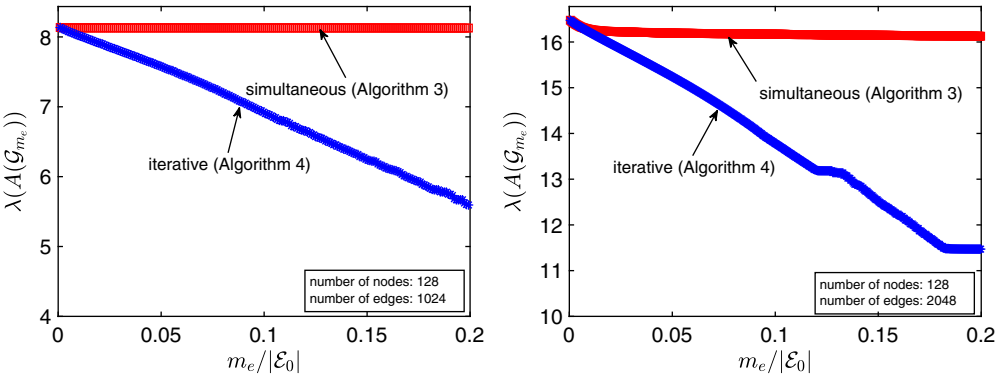

Figure 11. Comparison between simultaneous and iterative link removal strategies on nonring lattice network.

performance and communication cost of both strategies. Specifically, in general the iterative link removal strategy yields a better performance in comparison to simultaneous link removal strategy. Furthermore, the performance gap becomes larger as the number of links to be removed increases. However, the iterative link removal comes at a price of a higher communication cost in contrast to the simultaneous link removal strategy. As a future work, it is desirable to develop distributed estimation algorithms which converge in finite time in order to effectively apply the proposed idea to large-size networks.

https://doi.org/10.1017/nws.2022.25 Published online by Cambridge University Press

Funding. The work was supported by the Academy of Finland under academy project decision number 330073.

Competing interests. None.

References

Bishop, A. N., & Shames, I. (2011). Link operations for slowing the spread of disease in complex networks. *Europhysics Letters*, *95*, 18005.

Bullo, F. (2018). *Lectures on network systems* (1st ed.). Scotts Valley, CA: CreateSpace. With contributions by J. Cortes, F. Dorfler, and S. Martinez.

Chen, C., Tong, H., Prakash, B. A., Eliassi-Rad, T., Faloutsos, M., & Faloutsos, C. (2016). Eigen-optimization on large graphs by edge manipulation. *ACM Transactions on Knowledge Discovery from Data*, *10*(4), 49.

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations* (3rd ed.). Baltimore, MD: Johns Hopkins University Press.

Gusrialdi, A. (2021). Distributed algorithm for link removal in directed networks. In: R. M. Benito, C. Cherifi, H. Cherifi, E. Moro, L. M. Rocha, & M. Sales-Pardo (Eds.), *Complex networks & their applications ix* (pp. 505–521). Cham: Springer International Publishing.

Gusrialdi, A., & Qu, Z. (2017). Distributed estimation of all the eigenvalues and eigenvectors of matrices associated with strongly connected digraphs. *IEEE Control Systems Letters*, *1*(2), 328–333.

Gusrialdi, A., Qu, Z., & Hirche, S. (2019). Distributed link removal using local estimation of network topology. *IEEE Transactions on Network Science and Engineering*, *6*(3), 280–292.

Li, C., Wang, H., & Van Mieghem, P. (2013). Epidemic threshold in directed networks. *Physical Review E*, *88*(6), 062802.

Li, N., Zhang, N., & Das, S. (2011). Preserving relation privacy in online social network data. *IEEE Internet Computing*, *15*(3), 35–42.

Liu, J., & Anderson, B. D. O. (2020). Communication-efficient distributed algorithms for solving linear algebraic equations over directed graphs. In *Proceedings of the 59th iEEE conference on decision and control* (pp. 5360–5365).

McDaniel, P., & McLaughlin, S. (2009). Security and privacy challenges in the smart grid. *IEEE Security & Privacy Magazine*, *7*(3), 75–77.

Milanese, A., Sun, J., & Nishikawa, T. (2010). Approximating spectral impact of structural perturbations in large networks. *Physical Review E*, *81*(4), 046112.

Mou, S., Liu, J., & Morse, A. S. (2015). A distributed algorithm for solving a linear algebraic equation. *IEEE Transactions on Automatic Control*, *60*(11), 2863–2878.

Nejad, B. M., Attia, S. A., & Raisch, J. (2009). 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).

Prakash, B. A., Chakrabarti, D., Valler, N. C., Faloutsos, M., & Faloutsos, C. (2012). Threshold conditions for arbitrary cascade models on arbitrary networks. *Knowledge and Information Systems*, *33*(3), 549–575.

Shames, I., Charalambous, T., Hadjicostis, C. N., & Johansson, M. (2012). Distributed network size estimation and average degree estimation and control in networks isomorphic to directed graphs. In *Proceedings of annual allerton conference on communication, control, and computing* (pp. 1885–1892).

Van Mieghem, P., & Van de Bovenkamp, R. (2013). Non-markovian infection spread dramatically alters the susceptible-infected-susceptible epidemic threshold in networks. *Physical Review Letters*, *110*(10), 108701.

Van Mieghem, P., Stevanović, D., Kuipers, F., Li, C., van de Bovenkamp, R., Liu, D., & Wang, H. (2011). Decreasing the spectral radius of a graph by link removals. *Physical Review E*, *84*(Jul), 016101.

Wang, Y., Chakrabarti, D., Wang, C., & Faloutsos, C. (2003). Epidemic spreading in real networks: An eigenvalue viewpoint. In *Proceedings of the 22nd international symposium on reliable distributed systems* (pp. 25–34).

Wood, R. J., & O'Neill, M. J. (2004). An always convergent method for finding the spectral radius of an irreducible non-negative matrix. *Australian and New Zealand Industrial and Applied Mathematics Journal*, *45*, C474–C485.

Wu, Y., Zhang, T., Chen, S., & Wang, T. (2017). The minimum spectral radius of an edge-removed network: A hypercube perspective. *Discrete Dynamics in Nature and Society*, *2017*, 1–8.

Yang, X., Li, P., Yang, L.-X., Wu, Y., & Deng, Y. (2016). Reducing the spectral radius of a torus network by link removal. *Plos One*, *11*(5), e0155580.

A preliminary version of this paper appeared as Gusrialdi, A. (2021). Distributed Algorithm for Link Removal in Directed Networks. In International Conference Complex Networks & Their Applications (pp. 509–521). Cham: Springer. https://doi.org/10.1007/978-3-030-65347-7_42

Cite this article: Gusrialdi A. Connectivity-preserving distributed algorithms for removing links in directed networks. *Network Science* https://doi.org/10.1017/nws.2022.25