Inverse All Shortest Path Problem

Zhihao Qiu, Ivan Jokić, Siyu Tang, *member, IEEE*, Rogier Noldus, and Piet Van Mieghem, *Fellow, IEEE*

Abstract—Although resource management schemes and algorithms for networks are well established, we present two novel ideas, based on graph theory, that solve inverse all shortest path problem. Given a symmetric and non-negative demand matrix, the inverse all shortest path problem (IASPP) asks to find a weighted adjacency matrix of a graph such that all the elements in the corresponding shortest path weight matrix are not larger than those of the demand matrix. In contrast to many inverse shortest path problems that are NP-complete, we propose the Descending Order Recovery (DOR) that exactly solves a variant of IASPP, referred to as optimised IASPP. The network provided by DOR minimized the number of links and the sum of the link weights among all the graphs with the same shortest path weight matrix. Our second proposed algorithm, Omega-based Link Removal (OLR), solves the optimised IASPP by utilising the effective resistance from flow networks. The essence of our idea is the applications of properties of *flow networks*, such as electrical power grids, to compute the needed resources in *path networks* subject to end-to-end demands, such as telecommunication networks where quality of service constraints specify the end-to-end demands.

Index Terms—Complex network, Inverse all shortest path problem, Graph theory, Shortest path, Effective resistance.

1 Introduction

The design, dimensioning or operation of networks is often constrained by end-to-end limits. For example, a telephone call requires that the voice packets travel through a telecommunication network with a designated maximum latency; the delay between a source and a destination is limited to about 150ms. However, real-time control of systems over the Internet may require a lower end-to-end delay. Thus, different services (voice, video, ftp, email, etc.) typically require a different end-to-end delay. Usually, a telecom operator can determine the demand matrix D containing the maximum tolerably end-to-end delay d_{ij} between node i and node j in the network. However, given the demand matrix D, a telecom operator is still confronted to dimension the network, both topology and link weights, so that transport along the "best" path between any pair (i,j) of nodes consumes less time than the maximum tolerable end-to-end delay d_{ij}. Here, we focus on finding a solution to the operator's problem, which we call "inverse all shortest path problem"(IASPP). Other applications of IASPP are the design and construction of transportation networks, where the goal entails creating a network that ensures commute times between stations are constrained by specific upper bounds. Similar challenges occur in wireless sensor and actuator networks [1], mobile communication radio access networks [2], etc. An exploration of practical applications is discussed in Section 6.

While extensive research has focused on finding the shortest paths in a given graph, limited attention is given to the inverse direction, i.e. obtaining or recovering a graph based on the shortest path weights between each node pair as IASPP. A related challenge, termed the inverse shortest path problem (ISPP), which has garnered attention in prior research [3], [4], [5], [6], [7], [8], [9], is reviewed in Section 2. ISPP asks for making a set of predetermined paths in the graph the shortest paths, after modification and/or ensuring the shortest path weights between specific node pairs are bounded by given demands. Applications of the ISPP occur in the design of networks [3], [10], modelling traffic [5] and seismic tomography [3], [4]. However, in many practical scenarios, the topology of the network is unknown, rendering existing ISPP approaches inapplicable. In contrast to ISPP, our IASPP only requires a demand matrix as input. Additionally, the approach we propose in Section 3 not only furnishes a graph that satisfies specified demands, but also stands as an effective technique of "network sparsification" [11] and helps to better understand the importance of different links within a network.

Before introducing the inverse all shortest path problem (IASPP) in Section 2, we explain the terminology. We consider a graph G that possesses a set $\mathcal{N}$ of N nodes and a set $\mathcal{L}$ of L links. The graph G can be represented [12] by an $N \times N$ adjacency matrix A, with element $a_{ij} = 1$ if there is a link in G between node $i \in \mathcal{N}$ and node $j \in \mathcal{N}$, otherwise $a_{ij} = 0$. Each link $l \in \mathcal{L}$ has a weight w_l, which is a positive real number that specifies a property of the link, e.g. the resistance in an electrical graph or the delay when transmitting IP packets over that link. On the graph G, two different types of transport are possible that lead to either "path networks" or "flow networks". In a path network, the transport of items follows a single path $\mathcal{P}_{ij}$ between a node pair (i,j), whereas in a flow network, the transport from node i to node j propagates over all possible paths from node i to node j. Two typical examples are a communication network, where IP packets follow most of the time a single path $\mathcal{P}_{ij}$ from source i to destination j, and a power grid, where electrical current flows over all possible paths.

The weight $w(\mathcal{P}_{ij}) = \sum_{l \in \mathcal{P}_{ij}} w_l$ of a path $\mathcal{P}_{ij}$ between a node pair (i,j) consists [13] of the sum of the weights over

- *Z.Qiu, I.Jokić, and P. Van Mieghem are with Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2600 GA Delft, The Netherlands.*
  E-mail: Z.Qiu-1@tudelft.nl
- *R.Noldus is with Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2600 GA Delft, The Netherlands, and also with Ericsson, The Netherlands.*
- *S.Tang is with Huawei Munich Research Center, Riesstrasse 25, 80992 Munich, Germany.*

all links that belong to that path $\mathcal{P}_{ij}$. We will denote by $\mathcal{P}^*_{ij}$ the shortest path between a node pair (i, j). The shortest path $\mathcal{P}^*_{ij}$ minimizes the path weight over all paths $\mathcal{P}_{ij}$ and obeys $w\left(\mathcal{P}^*_{ij}\right) \leq w\left(\mathcal{P}_{ij}\right)$. In most real-world networks, there is only one shortest path $\mathcal{P}^*_{ij}$, but, in general, there can be many shortest paths between the same node pair (i, j), in particular in unweighted graphs, where each link has the same link weight[1], i.e. $w_{ij} = w$ for all elements of the $N \times N$ link weight matrix W. The weighted adjacency matrix is $\widetilde{A} = W \circ A$, where the Hadamard product $\circ$ means a direct elementwise multiplication, $\widetilde{a}_{ij} = w_{ij}a_{ij}$ and we use "tilde" notation for weighted graph matrices[2]. In our setting, $\widetilde{a}_{ij} = 0$ means that there is no link between node i and node j, because we exclude zero link weights, i.e. $w_{ij} > 0$, as in Dijkstra's shortest path algorithm [14], [16], [17] and in order to avoid the complication that a zero weight, i.e. $w_{ij} = 0$, would physically mean that node i and j are the same. The separation between link weights, represented by the link weight matrix W, and underlying graph G, represented by the adjacency matrix A, is obvious in unweighted graphs, where $W = wJ$ and $J = u \cdot u^T$ is the all-one matrix and u is the all-one vector. In the unweighted case, the graph is confining. In the other extreme, where link weights are highly variable and where the minimum link weight $w_{\min} > 0$ is orders of magnitude smaller than the maximum link weight $w_{\max}$, the underlying graph G is less confining than the link weight structure[3], which effectively thins out the graph. Indeed, mainly links with small link weights are relevant in a shortest path problem and large link weights may be ignored[4] from the onset, especially if link weights are assigned per link independently of the other links (see also [13, Chapter 16], [18], [19], [20]). In a shortest path setting, links with low link weights are generally more costly than links with high link weights.

Let v_k denote the potential or voltage of node k in the graph G. The effective resistance ω_{ij} between node i and node j equals the voltage difference $\omega_{ij} = \frac{v_i - v_j}{I_c}$ when a unit current $I_c = 1$ Ampere is injected in node i and leaves the network at node j. The $N \times N$ effective resistance matrix Ω with elements ω_{ij}, studied in e.g. [12], [21], [22], [23], [24] and [12, Chapt. 5], is briefly reviewed in Sec. 1.2. If the graph G is connected[5], then the effective resistance ω_{ij} as well as the path weight $w\left(\mathcal{P}_{ij}\right)$ is finite for any node pair (i, j) and a shortest path $\mathcal{P}^*_{ij}$ exists between each node pair (i, j). We define the $N \times N$ matrix S, that contains all shortest path weights with element $s_{ij} = w\left(\mathcal{P}^*_{ij}\right)$. If the weighted adjacency matrix $\widetilde{A}$ is known, then the matrix S is readily found via a shortest path algorithm, like Dijkstra's shortest path algorithm. Dijkstra's shortest path computation is very efficient and only requires $O\left(N \log N\right)$ elementary operations. Both the effective resistance matrix Ω and the shortest path weight matrix S are distance matrices[6].

In the sequel, we limit ourselves to connected, undirected, simple[7] graphs. Consequently, the $N \times N$ symmetric matrices A, W, $\widetilde{A}$, Ω and S are non-negative with zero diagonal elements.

The main contributions of this work are as follows:

1) We propose a novel problem named "Inverse all shortest path problem" (IASPP) and its variant "the optimised IASPP" (OIASPP). The IASPP asks for a weighted graph whose shortest path weight between each node pair satisfies a given demand.
2) We prove that OIASPP is not NP-complete.
3) We propose the Descending Order Recovery (DOR) algorithm that exactly solves OIASPP. The DOR graph minimizes the number of links and the sum of the link weights among all the graphs with the same shortest path weight matrix.
4) We demonstrate that DOR is also an effective network sparsification algorithm.
5) We propose the Omega-based Link Removal (OLR) algorithm, which solves OIASPP by utilising the effective resistance [12, Chapter 5]. OLR invokes properties of *flow* networks, such as electrical power grids, to compute the needed resources in *path* networks subject to end-to-end demands, such as telecommunication networks.
6) We discuss the applications of IASPP and evaluate the performance of DOR and OLR.

The paper is outlined as follows. In Section 1.1 and 1.2, we introduce notations to describe IASPP. We formally define IASPP and its variant OIASPP in Section 2 and review related problems from literature. In Section 3 and Section 4, we respectively propose two algorithms, DOR and OLR, to solve the optimised inverse all shortest path problem (OIASPP). Section 5 compares and evaluate the proposed algorithms by simulations. Section 6 introduces the potential applications of IASPP. Finally, we summarise our results in Section 7.

1.1 The Laplacian matrix Q

The $N \times 1$ degree vector $d = A \cdot u$ contains the degree d_i of each node i and the corresponding diagonal matrix $\Delta = \text{diag}(d)$ has the nodal degrees on its main diagonal. The eigenvalue decomposition of the $N \times N$ Laplacian $Q = \Delta - A$,

$$Q = Z \cdot \text{diag}(\mu) \cdot Z^T, \tag{1}$$

defines the set of N orthogonal $N \times 1$ eigenvectors z_i contained in columns of the $N \times N$ eigenvector matrix Z, and the set of N eigenvalues $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_N$. Due to double orthogonality of the eigenvector matrix Z (i.e.

1. The shortest path does not change if all weights are multiplied by a constant $\alpha > 0$.

2. The flow network is characterized by the subscript F, i.e. $\widetilde{A}_F$ is the weighted adjacency matrix of a flow network, while $\widetilde{A}$ denotes the weighted adjacency matrix of a path network.

3. The link weight structure refers to the entire ensemble $\{w_l\}_{l\in\mathcal{L}}$ of all link weights in the graph as one coherent set, possibly generated by a process that takes correlations of weights over links into account. The matrix W can then be considered as one particular realization of the link weight structural process.

4. If their removal does not disconnect the graph.

5. The weighted adjacency matrix $\tilde{A}$ is called irreducible when the graph G is connected (see [13, p. 183]; [12, art. 167 on p. 235]). For a connected graph, the (weighted) Laplacian only has 1 zero eigenvalue and its rank is $N - 1$.

6. Any element h_{ij} of a distance matrix H is non-negative $h_{ij} \geq 0$, but $h_{ii} = 0$ and h_{ij} obeys the triangle inequality: $h_{ij} \leq h_{ik} + h_{kj}$.

7. A simple graph has no multiple links between a same pair of nodes and also no self-loops, i.e. $a_{ii} = 0$ for each node $i \in \mathcal{N}$.

$Z \cdot Z^T = I$ and $Z^T \cdot Z = I$), where I is the $N \times N$ identity matrix, (1) can be transformed into a weighted sum of N outer vector products

$$Q = \sum_{i=1}^{N} \mu_i \cdot z_i \cdot z_i^T. \tag{2}$$

As of any real, symmetric matrix [12], the eigenvalues of Laplacian Q are real and non-negative because Q is a positive semidefinite matrix. From $Q \cdot u = 0$, we observe that $\mu_N = 0$ and $z_N = u$ and thus $\det Q = 0$. Consequently, the Laplacian Q is not invertible. However, the pseudoinverse[8] of the Laplacian [23]

$$Q^{\dagger} = \sum_{i=1}^{N-1} \frac{1}{\mu_i} \cdot z_i \cdot z_i^T \tag{3}$$

obeys $Q^{\dagger} \cdot Q = Q \cdot Q^{\dagger} = I - \frac{1}{N} \cdot J$. In this work we consider a weighted graph G, where a link l between node i and node j is defined by its weight

$$w_{ij} = w_l = \frac{1}{r_l},$$

with r_l denoting link l resistance and $r_l > 0$, $w_l > 0$.

1.2 Effective Resistance

The effective resistance ω_{ij} between node i and node j is defined as [12]

$$\omega_{ij} = (e_i - e_j)^T \cdot Q^{\dagger} \cdot (e_i - e_j), \tag{4}$$

where the $N \times 1$ basic vector e_i has only one non-zero element $(e_i)_i = 1$. The effective resistance ω_{ij} quantifies the dissipated power when the current of 1 Ampere is applied between the nodes i and j. The equation in (4) can be transformed into a matrix form, defining the $N \times N$ effective resistance matrix

$$\Omega = \zeta \cdot u^T + u \cdot \zeta^T - 2 \cdot Q^{\dagger}, \tag{5}$$

where the $N \times 1$ vector $\zeta = \left(Q^{\dagger}_{11},\, Q^{\dagger}_{22},\, \ldots,\, Q^{\dagger}_{NN}\right)$ contains the diagonal elements of the pseudoinverse of the Laplacian Q. The effective resistance ω_{ij} between directly connected nodes i and j (i.e. $a_{ij} = 1$), represents the effective resistance of a parallel connection

$$\frac{1}{\omega_{ij}} = \frac{1}{r_{ij}} + \frac{1}{(\omega_{G^*})_{ij}} \tag{6}$$

between the resistance of a direct link r_{ij} and the effective resistance $(\omega_{G^*})_{ij}$ between nodes i and j in the graph $G^* = G \setminus l_{ij}$, where the link l_{ij} is removed.

Lemma 1. A link $l_{ij} \in \mathcal{L}$ of a graph $G(\mathcal{N}, \mathcal{L})$ connects two disconnected sub-graphs G_1 and G_2, i.e. $\mathcal{L}(G_1) \cup \mathcal{L}(G_2) \cup l_{ij} = \mathcal{L}(G)$ and $\mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \emptyset$ if and only if it holds

$$\omega_{ij} = r_{ij}.$$

Proof: When link l_{ij} of a graph G connects two disconnected sub-graphs G_1 and G_2, the effective resistance of a graph $G^* = G \setminus l_{ij}$ equals $r^*_{ij} = \infty$. Therefore, (6) transforms into $\omega_{ij} = r_{ij}$. □

The effective resistance ω_{ij} between adjacent nodes i and j is upper bounded by the resistance r_{ij} of the direct link between them

$$\omega_{ij} = \frac{r_{ij} \cdot (\omega_{G^*})_{ij}}{r_{ij} + (\omega_{G^*})_{ij}} \leq \min\left(r_{ij},\, (\omega_{G^*})_{ij}\right).$$

Otherwise, if $a_{ij} = 0$, then the effective resistance ω_{ij} is upper bounded by the sum of resistances of links forming the shortest path between the nodes. In both cases, if more paths exist connecting two nodes, then there are more possible paths for the current to flow simultaneously and thus, the effective resistance lowers. The sum of all elements of the $N \times N$ effective resistance matrix Ω defines the effective graph resistance [12]

$$R_G = \frac{1}{2} \cdot u^T \cdot \Omega \cdot u = N \cdot \sum_{i=1}^{N-1} \frac{1}{\mu_i}. \tag{7}$$

2 Inverse all shortest path problem

2.1 Statements of inverse all shortest path problems

Problem 1 (Inverse All Shortest Path Problem (IASPP)). Given an $N \times N$ symmetric demand matrix D with zero diagonal elements but positive off-diagonal elements. Determine an $N \times N$ weighted adjacency matrix $\widetilde{A}$, such that the corresponding shortest path weight matrix S obeys[9] $S \preccurlyeq D$

Since an element in the shortest path weight matrix S can be any positive number by scaling the weighted adjacency matrix, the IASPP generally has infinitely many solutions. Therefore, optimisation criteria such as the minimization of a norm $||D - S||$ are added. An instance [10] of IASPP is the optimised inverse shortest path problem (OIASPP).

Problem 2 (Optimised Inverse All Shortest Path Problem (OIASPP)). Given an $N \times N$ symmetric demand matrix D with zero diagonal elements but positive off-diagonal elements. Determine an $N \times N$ weighted adjacency matrix $\widetilde{A}$, such that the corresponding shortest path weight matrix S obeys $S \preccurlyeq D$ and minimizes a norm $||D - S||$.

Van Mieghem [10] demonstrated that any demand matrix D can be transformed into a distance matrix D' with $D' \preccurlyeq D$, where D' represents the (modified) demand matrix that is also a distance matrix: If $d_{ik} + d_{kj} < d_{ij}$ for at least one node $k \in \mathcal{N}$ which violates the triangle inequality of a distance matrix, then we can replace $d_{ij} = \min_{1 \leq k \leq N}(d_{ik} + d_{kj})$ and $d_{ji} = d_{ij}$. In the following, we assume that the demand matrix D is also a distance matrix. A complete graph whose weighted adjacency matrix $\widetilde{A} = D$ is a solution of the OIASPP with demand matrix D. Consequently, given a demand matrix D, we can obtain at least one solution of the OIASPP. In 1965, Hakimi and Yau [25] proved that if a weighted graph G is an N-node realization of an $N \times N$ distance matrix D', i.e. the corresponding shortest path weight matrix S of G equals D', and there does not exist three nodes i, j and k such

8. We restrict the analysis to connected simple graphs, as the number of zero eigenvalues of Laplacian Q equals the number of connected components in a graph. More precisely, (3) does not hold in the case of a disconnected graph.

9. The notation $\preccurlyeq$ is used for componentwise inequality, i.e. $S \preccurlyeq D$ means that $s_{ij} \leq d_{ij}$ for each $i = 1, 2, \ldots, N$ and each $j = 1, 2, \ldots, N$.

that $w_{ij} > s_{ik} + s_{kj}$, where w_{ab} is the link weight between nodes a and b and s_{ab} denotes the shortest path weight, then G is unique. Hence, if there is only one solution of the OIASPP, then the resulting graph is a complete graph [10] and $w_{ij} \leq s_{ik} + s_{kj}$ holds for arbitrary three nodes i, j and k, when the graph size $N \geq 3$. When the demand matrix D is a shortest path weight matrix generated by a tree, Van Mieghem [10] has solved OIASPP exactly as explained in Section 2.2.

In this paper, we focus on general underlying graphs rather than trees or complete graphs. We respectively propose two algorithms Descending Order Recovery (DOR) and Omega-based Link Removal (OLR) to solve OIASPP in Section 3 and Section 4. Since the computational complexity of DOR is polynomial, we have incidentally proved that OIASPP is not NP-complete.

2.2 Literature review

Before investigating IASPP, we explain the related inverse shortest path problem (ISPP). Both ISPP and IASPP are "inverses" of the shortest path problem, that ask for a graph given the shortest paths or shortest path weights between node pairs. However, ISPP requires both the shortest paths (or shortest path weights) *and* the original graph, while IASPP only necessitates a demand matrix, that specifies the maximum shortest path weights, as input.

Problem 3 (Inverse Shortest Path Problem (ISPP)). Given an $N \times N$ weighted adjacency matrix $\widetilde{A}$ with link weight matrix W and a set of paths $\{\mathcal{P}_{ij}\}$. Determine an $N \times N$ non-negative link weight matrix W' and the corresponding graph H such that all the paths $\mathcal{P}_{ij}$ belonging to $\{\mathcal{P}_{ij}\}$ are the shortest paths in the obtained graph H.

We will introduce several representative generalizations or variants of ISPP below.

In 1992, Burton and Toint [3] proposed a quadratic programming algorithm based on the Goldfarb-Idnani method [26] to solve a variant of ISPP, which we denote by $\text{ISPP}_{\text{Burton}}$:

Problem 4 (Inverse Shortest Path Problem Burton (ISPP$_{\text{Burton}}$)). Given an $N \times N$ weighted adjacency matrix $\widetilde{A}$ with link weight matrix W and a set of paths $\{\mathcal{P}_{ij}\}$. Determine an $N \times N$ non-negative link weight matrix W' and the corresponding graph H such that all the paths $\mathcal{P}_{ij}$ belonging to $\{\mathcal{P}_{ij}\}$ are the shortest path in the obtained graph H and minimize $||W' - W||$.

Burton and Toint utilised l_2 norm $||W' - W|| = \sqrt{\sum_i \sum_j (w'_{ij} - w_{ij})^2}$, where w'_{ij} and w_{ij} represent the elements of W' and W respectively. A specialized Goldfarb-Idnani method can then be implied. The approach involves iterative adjustments to the matrix W', leading to the eventual weighted graph H, in which $\mathcal{P}_{ij}$ belonging to the given path set $\{\mathcal{P}_{ij}\}$ are the shortest paths. The method works in both directed and undirected graphs.

Different variants and modified methods following $\text{ISPP}_{\text{Burton}}$ are discussed in [5], [6], [7], [8]. In 1999, Fekete et al. [9] considered a more general ISPP, where only the shortest path weight between pairs of nodes is given, but not the paths achieving them. Given a graph G with adjacency matrix A and a demand matrix D, $\text{ISPP}_{\text{Fekete}}$ aims to find a "weight function" of the weighted adjacency matrix $\widetilde{A}$ such that the demand matrix D is exactly the shortest path weight matrix S, where the weight function describes all the weighted adjacency matrices whose corresponding shortest path weight matrix $S = D$. The demand matrix D in $\text{ISPP}_{\text{Fekete}}$ must be a distance matrix measuring the shortest path weight between several pairs of nodes in graph G. *Not* all the pairs of nodes in graph G are necessarily included in the demand matrix D. Fekete et al. [9] proved that $\text{ISPP}_{\text{Fekete}}$ is NP-complete by reducing $\text{ISPP}_{\text{Fekete}}$ to a vertex-disjoint paths problem.

All mentioned variants of ISPP require the original weighted adjacency matrix $\widetilde{A}$ or adjacency matrix A. In contrast, Hakimi and Yau [25] investigated a "weighted graph realization" with only an $N \times N$ demand matrix D as input, which is also a distance matrix. Hakimi and Yau [25] presented an algorithm to obtain a graph H on N' nodes by adding $N' \geq N$ nodes into the graph such that the corresponding shortest path weight matrix $S = D$. If we extract the shortest path weights between node pairs that belonging to the first N nodes and form a shortest path weight matrix S, then $S = D$. Since the input in [25] contains all the shortest path weights in a graph, we call the problem "inverse all shortest path problem" (IASPP).

If the given distance matrix D can be realized by a tree t, Van Mieghem [10] proposed an elegant algorithm to recover the tree t from D by exploiting the analogy between flow networks and path networks. For undirected flow networks, Fiedler [27], [28] has presented the following block matrix relation,

$$\begin{pmatrix} 0 & u^T \\ u & \Omega \end{pmatrix}^{-1} = \begin{pmatrix} -2\sigma^2 & p^T \\ p & -\frac{1}{2}\widetilde{Q} \end{pmatrix} \tag{8}$$

with $\Omega p = 2\sigma^2 u$, where $\widetilde{Q} = \widetilde{\Delta}_F - \widetilde{A}_F$ is the weighted Laplacian matrix of a flow network and the diagonal matrix $\widetilde{\Delta}_F = \text{diag}(A_F u)$, $\widetilde{A}_F$ is the weighted adjacency matrix of a flow network, the variance $\sigma^2 = \frac{\zeta^T \widetilde{Q} \zeta}{4} + R_G$, where R_G is the effective graph resistance [23] and u is $N \times 1$ the all-one vector. The vector ζ contains the diagonal elements of pseudoinverse $Q^\dagger$ of the Laplacian $\widetilde{Q}$. Specifically, Van Mieghem [10] defined the weight of a link $w_{ij} = r_{ij}$ as the resistance (in Ohm), then the weighted Laplacian $\widetilde{Q}$ has non-zero elements $\widetilde{q}_{ij} = -\frac{1}{r_{ij}}$ and $\widetilde{a}_F = \frac{1}{r_{ij}}$ for $i \neq j$, where $\widetilde{a}_{ij} = r_{ij}$, but $(\widetilde{a}_F)_{ij} = (\widetilde{a})_{ij} = 0$ for $i \neq j$ if there is no link between node i and node j. The diagonal elements $(\widetilde{a}_F)_{ii} = (\widetilde{a})_{ii} = 0$ are always zero. Fiedler's block matrix relation (8) indicates a one-to-one relation between the effective resistance matrix Ω and the weighted Laplacian $\widetilde{Q}$ and therefore, also between the effective resistance matrix Ω and the weighted adjacency matrices $\widetilde{A}_F$ and $\widetilde{A}$. By applying block inverse formulae [12] to Fiedler's block matrix relation, it is shown in [10] that

$$2\sigma^2 = \frac{1}{u^T \Omega^{-1} u} \tag{9}$$

$$p = \frac{1}{u^T \Omega^{-1} u} \Omega^{-1} u \tag{10}$$

and the inverse of the effective resistance matrix is

$$\Omega^{-1} = \frac{1}{2\sigma^2} p p^T - \frac{1}{2}\widetilde{Q} \tag{11}$$

Hence, with $\widetilde{Q} = \widetilde{\Delta}_F - \widetilde{A}_F$, the weighted adjacency matrix follows as

$$\widetilde{A}_F = \widetilde{\Delta}_F + 2\Omega^{-1} - \frac{1}{\sigma^2} pp^T. \quad (12)$$

If the graph G is a tree, then the shortest path matrix S equals the effective resistance matrix Ω, because there exists exactly one path in a tree between each pair of nodes [22]. The weighted adjacency matrix $\widetilde{A}_F$ can be deduced from (12) by replacing Ω by S. Hence, the weighted adjacency matrix $\widetilde{A}$ follows by taking the element-wise inverse of $\widetilde{A}_F$. The zero elements in $\widetilde{A}_F$ should not be inverted, but should instead be transferred to $\widetilde{A}$. Indeed, the obtained $\widetilde{A}$ is an exact solution of the OISPP for any tree: If the given demand matrix D is a distance matrix such as the shortest path weight matrix S, then the weighted adjacency matrix $\widetilde{A}$ can be obtained from (12) with $\Omega = S$. We call this method the "flow analogue method".

As explained in Appendix B, the algebraic flow analogue method is hard to extend from a tree graph to a general graph. In the sequel, we solve OIASPP for general graphs.

3 Descending Order Recovery algorithm

In this section, we propose the Descending Order Recovery algorithm (DOR) that solves OIASPP exactly. If a demand matrix D is the shortest path weight matrix S of an arbitrary graph G, then DOR retrieves the graph H satisfying the norm $||D - S'|| = 0$, where S' is the shortest path weight matrix of H. For a given demand matrix D, the graph H obtained by DOR is unique and reaches a minimum number of links and a minimum sum of the link weights among all OIASPP solutions with the same demand matrix D. The resulting graph H generally has less links than graph G. If graph G is unweighted, the resulting graph H is the same graph as G.

3.1 Properties of DOR

Our main idea of DOR is:

1) Given a demand matrix D, find the minimum spanning tree of the complete graph G_D, whose weighted adjacency matrix $\widetilde{A} = D$;
2) Add a link between two nodes i and j whose shortest path weights $s_{ij} > d_{ij}$, with link weight $w_{ij} = d_{ij}$;
3) Repeat 2 until $s_{ij} \le d_{ij}$ for each $i, j \in \mathcal{N}$.

If we remove a link l in graph G and obtain graph H, then the link l either (a) belongs or (b) does not belong to the shortest path $\mathcal{P}^*_{ij}$ between two nodes i and j. In case (a), removing link l does not change the shortest path $\mathcal{P}^*_{ij}$ nor the shortest path weight s_{ij}. In case (b), the shortest path between nodes i and j is changed, but the shortest path weight s'_{ij} in H cannot be smaller than s_{ij}, otherwise, the shortest path weight s'_{ij} would also be the shortest path weight in G. Thus, the shortest path weight $s_T(ij)$ between arbitrary nodes i and j in the minimum spanning tree T of a graph G is not smaller than the shortest path weight $s_G(ij)$ in the graph G and step 1 ensures that the lower bound of the shortest path weight matrix of our obtained graph H is D. The upper bound of the shortest path weight matrix of the obtained graph is also D after performing step 2 and 3. DOR obtains a graph H satisfying the norm $||D - S'|| = 0$. We present the pseudo code of DOR initialised with a minimum spanning tree as Algorithm 4 in Appendix F.

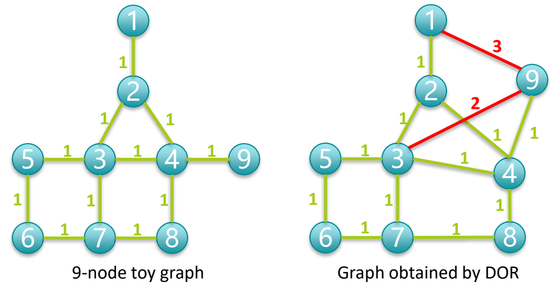


Fig. 1. Visualization of redundant links in the graph obtained by DOR. We generate a 9-node toy graph and obtain the corresponding shortest path weight matrix as the demand matrix D. A graph is then obtained by DOR with the demand matrix D as input. The redundant links are highlighted.

The graph H obtained by DOR may have more links than the original graph G. In the worst case, the resulting H is a complete graph, whose weighted adjacency matrix equals the demand matrix D. We call l_{ij} a "redundant" link if we can find another node k, besides i and j, in the graph such that $\widetilde{a}_{ij} = w_{ij}a_{ij} \ge s_{ik} + s_{kj}$. For example, as shown in Fig. 1, link l_{19} can be replaced by l_{12}, l_{24} and l_{49} when calculating the shortest path between nodes 1 and 9 in the graph obtained with DOR, since $\widetilde{a}_{19} = s_{14} + s_{49}$. Removing link l_{19} would not change the shortest path weight matrix S nor the connectivity of the original graph.

Algorithm 1 Descending Order Recovery (DOR)

Input: $N \times N$ demand matrix $D = S$: a shortest path weight matrix of a graph G
Output: $N \times N$ weighted adjacency matrix $\widetilde{A}$
1: $\widetilde{A} \leftarrow D$ and $\widetilde{A}$ specifies graph G
2: $\forall$ positive link weights in G and any node $k \ne i, j$
3: **if** $\widetilde{a}_{ij} \ge s_{ik} + s_{kj}$ **then**
4: $\widetilde{a}_{ij} \leftarrow 0, \widetilde{a}_{ji} \leftarrow 0$
5: **end if**
6: **return** $\widetilde{A}$

If a link l_{ij} is redundant in a weighted adjacency matrix $\widetilde{A}$ obtained by DOR, i.e. if there exists a node k such that $s_{ik} + s_{kj} \le \widetilde{a}_{ij}$, we then remove the link between nodes i and nodes j and let $a_{ij} = 0$. Hakimi and Yau [25] proved that there is only one graph which does not have redundant links among all the graphs with the same shortest path weight matrix S. Therefore, the graph H obtained by DOR is unique for a given demand matrix D after removing all redundant links. Hence, DOR can be further simplified: After removing all redundant links in the complete graph G_D whose weighted adjacency matrix equals the demand matrix D, we obtain the solution graph H, which solves OIASPP exactly. The pseudo code for simplified DOR is shown in Algorithm 1.

Property 1. Given a demand matrix D, the obtained graph H by DOR reaches a minimum number of links among all the OIASPP solutions.

Proof: By contradiction: Suppose that there exists a graph H' such that the corresponding shortest path weight matrix $S = D$ and the graph H' has fewer links than the graph H obtained by DOR. The graph H' should have redundant links because both graph H' and graph H have the same shortest path weight matrix S and the graph H does not have redundant links. In that case, we construct a graph H'' by removing redundant links in graph H'. Since removing redundant links does not change the corresponding shortest path weight matrix, graph H'' and graph H have the same shortest path weight matrix S and do not have redundant links, which is impossible because there is only one graph that does not have redundant links among all the graphs with the same shortest path weight matrix S. Hence, the obtained graph H by DOR reaches a minimum number of links among all the solutions to an OIASPP given a demand matrix D. □

Because the graph H obtained by DOR minimizes the number of links among all the graphs with the same shortest path weight matrix S, we can only obtain a graph H' that has the same shortest path weight matrix S by adding redundant links. We thus have:

Property 2. Given a demand matrix D, the obtained graph H by DOR reaches a minimum sum of the link weights among all OIASPP solutions.

Given a demand matrix D that is a shortest path weight matrix S of an arbitrary "original" graph G. While the shortest path weight matrix S of the graph H obtained by DOR is identical to the shortest path weight matrix of the original graph G, the two graphs H and G themselves may not be the same. Specifically, when the demand matrix D is computed from an unweighted graph G, fortunately, we can remove all the redundant links by removing links whose weights are larger than 1 in the complete graph G_D. Since all the link weights in unweighted graphs are exactly 1, the shortest path weight s_{ij} between two nodes equals 1 if and only if nodes i and j are neighbours. Thus the adjacency matrix A of the graph after removing redundant links in the complete graph G_D is precisely the same as the adjacency matrix of the original unweighted graph G.

3.2 Examples

In Fig. 2, we respectively examine the number of links L_G and L_H of the original graph G and the DOR graph H. For each simulation, we generate an Erdős–Rényi (ER) random graph $G_p(N)$, where N is the number of nodes and p is the probability of connecting two nodes. The link weights of the ER graph $G_p(N)$ are uniformly distributed in $(0, 1)$. The $N \times N$ shortest path weight matrix S is calculated and equal to the demand matrix D. For different N and p, 1000 iterations are carried out.

Fig. 2 illustrates that DOR produces graphs H with fewer links than the original ER graphs G, provided the link density p is sufficiently large. An interesting phenomenon is that the resulting graph H seems to have a similar number of links, irrespective of the number L_G of links in the original graph. Hence, for a dense original graph G, DOR provides a sparser graph with the same shortest path weight matrix, but with a different adjacency matrix A, which can be regarded as "network sparsification" [11] that preserves all shortest path weights.

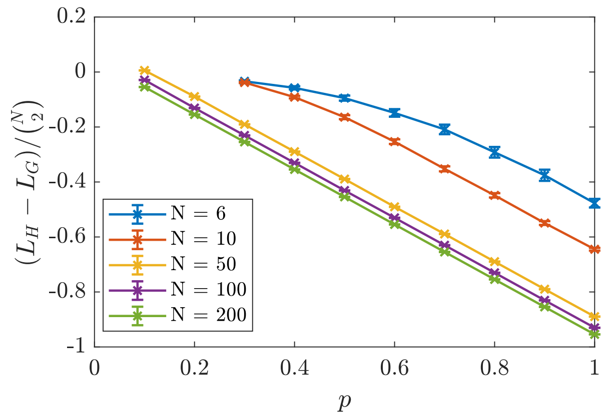


Fig. 2. The differences of number of links between the original graph and the graph obtained by DOR.

An instance of network sparsification is investigated by Simas, et al. [29]. Given a graph G with weighted adjacency matrix $\widetilde{A}$, Simas, et al. [29] focuses on obtaining a graph H, which they call the "distances backbone", with the same shortest path weight matrix S of graph G, but fewer links. The main idea is that the off-diagonal elements of the resulting weighted adjacency matrix $\widetilde{A}'$ are computed by

$$\begin{cases} \widetilde{a}'_{ij} = s_{ij} & \text{if } \widetilde{a}_{ij} = s_{ij} \\ \widetilde{a}'_{ij} = 0 & \text{if } \widetilde{a}_{ij} > s_{ij} \end{cases} \tag{13}$$

for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, N, j \neq i$. However, the method proposed by Simas et al. [29] always includes the redundant links such that $w_{ij} = s_{ik} + s_{kj}$, where w_{ij} is the weight of link l_{ij}. Thus, DOR can return a sparser graph than the distances backbone.

Van Mieghem and Wang [20] investigated the union of all shortest path trees $G_{\cup_{spt}}$, where the shortest path tree (SPT) rooted at some node is the union of the shortest paths from that node to all the other nodes. If a link l_{ij} is the shortest path $\mathcal{P}^*_{ij}$ between i and j, then l_{ij} must belong to the $G_{\cup_{spt}}$, because the $G_{\cup_{spt}}$ is the union of shortest paths between all possible source and destination nodes [20]. All the links in the graph H obtained by DOR belong to at least one shortest path $\mathcal{P}^*_{ij}$ and the graph H thus belongs to the $G_{\cup_{spt}}$. The inverse does not hold, because the union $G_{\cup_{spt}}$ may have redundant links l_{ij} in which $w_{ij} = s_{ik} + s_{kj}$.

3.3 Computational complexity of DOR

For each possible link l_{ij}, DOR determines whether the link is redundant by comparing the link weight w_{ij} with the sum of the shortest path weights $s_{ik}+s_{kj}$, where $k \in \mathcal{N}$ is a node different from node i and j. Hence, each link l_{ij} needs to be compared with the sum of the shortest path weights $s_{ik}+s_{kj}$ for $N-2$ nodes k in the worst case. The computational complexity of the worst case of DOR (Algorithm 1) is $O(N^3)$, because the demand matrix $D = O(N^2)$. OIASPP is thus not NP-complete! The main differences between OIASPP and three NP-complete variants of ISPP introduced in Section 2 and Appendix E lie in the given constraints. While the three NP-complete variants of ISPP restrict the resulting graph to a predetermined graph topology, OIASPP can be solved by changing both topology and link weights to meet the given constraints about shortest path weights.

4 Omega-based Link Removal Algorithm

The Omega-based Link Removal (OLR) algorithm recovers an as sparse as possible graph, with elements of the shortest path weight matrix $s_{ij} \in [bd_{ij}, d_{ij}]$, where d_{ij} is the given demand and $b \in [0, 1]$ is an input parameter. OLR leverages information captured by the effective resistance between pairs of nodes. Equation (6) enables us to determine the impact on the effective resistance between two neighbouring nodes when the shared link between them, denoted as l_{ij}, is eliminated. By targeting the removal of the link with the highest value of $\frac{1}{(\omega_{G^*})_{ij}}$, we achieve the smallest possible increase in the effective graph resistance R_G of the network. To enhance the efficacy of this approach for solving OIASPP, we introduce a refinement, which involves scaling the quantity $\frac{1}{(\omega_{G^*})ij}$ by the difference between the provided upper bound d_{ij} and the current shortest path weight s_{ij} for the pair of nodes (i, j). This strategic adjustment allows us to combine insights from the effective resistance measurements and the upper bound values supplied by the $N \times N$ demand matrix D.

The shortest path weight between two nodes is the sum of the link weights (i.e. corresponding elements of the weighted adjacency matrix $\tilde{A}$) belonging to that path. On the contrary, a link weight in a "flow network "(defined by the adjacency matrix $\tilde{A}_F$) has a dimension of the inverse of the resistance. Therefore, to utilise the analogy between shortest paths and effective resistance, we additionally define the $N \times N$ link weight matrix $\hat{W}$ containing the inverse link weights[10]

$$\hat{w}_{ij} = \begin{cases} \frac{1}{\tilde{a}_{ij}} & \text{if } \tilde{a}_{ij} > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{14}$$

where $i, j \in \mathcal{N}$. The corresponding $N \times N$ effective resistance matrix computed with $\hat{W}$ instead of $\tilde{A} = A \circ W$ is denoted as $\hat{\Omega}$.

In Algorithm 2, we propose an iterative algorithm that solves the IASPP problem by invoking the effective resistance between pairs of nodes. The OLR algorithm is initialised in line 1 by the complete graph with the adjacency matrix $A = J - I$, while the link weights equal (line 2) the corresponding shortest path weights in the demand matrix $D = S$, scaled by the input parameter b,

$$\widetilde{A} = b \cdot (A \circ D),$$

which ranges between 0 and 1. Link weights are scaled in line 2 for two reasons. Assume the demand matrix D is derived from an original graph. In case $b = 1$, if the proposed OLR algorithm recovers the exact topology as in the original graph G, then the link weights would also be the same. In general, OLR ensures the shortest path weight between directly connected nodes to be equal to the corresponding element of the provided upper bound in D, scaled by the input parameter b. Therefore, $b < 1$ allows OLR to achieve sparser graphs even from the original graph G, at the cost of increased norm [11] of $||D-S||$, still satisfying the bound $S \preccurlyeq D$. To determine which link should be removed in each iteration, in line 7 we compute the $N \times N$ matrix

$$R = \left(\hat{\Omega} - \hat{W}\right) \circ (D - S) \circ A,$$

where the $N \times N$ inverse link weight matrix $\hat{W}$ contains inverse link weights, as defined in (14). whose elements are dimensionless and denote the inverse effective resistance $\left(\hat{\Omega} - \hat{W}\right)_{ij}$ between a pair of neighbouring nodes (i.e. $a_{ij} = 1$), in case the direct link between them is removed (as in (6)), multiplied by the gap $(d_{ij} - s_{ij})$ between the shortest path weight between them and the given upper bound in D. We remove the existing link with the highest value in R (line 8), because the adjacent nodes are easily reachable via the rest of the graph when the link is removed, and the margin between the current shortest path weight and the upper bound is relatively high. After updating the adjacency matrix A (line 9), we redistribute the link weights (line 10) as $\widetilde{A} = b \cdot (A \circ D)$ and update (line 11) the $N \times N$ shortest path weight matrix S.

Link removal is performed until at least one shortest path weight in the obtained graph H exceeds the given upper bound in the $N \times N$ demand matrix D. At that point, the last removed link is returned (line 14), while the $N \times N$ weighted adjacency matrix $\widetilde{A}$ is provided as output.

OLR initialises the topology with a complete graph and iteratively removes links until at least one upper bound on the shortest path weight between node pairs is exceeded. In general, OLR can return any connected topology, even a

Algorithm 2 Omega-based Link Removal (OLR)

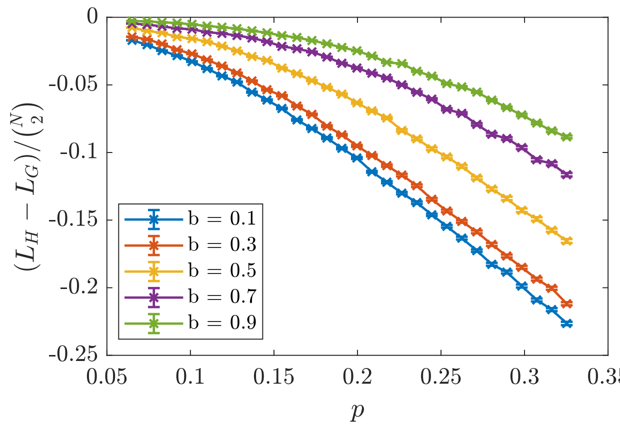
Input: $N \times N$ demand matrix $D = S$: a shortest path weight matrix of a graph G; input parameter $b \in [0, 1]$
Output: $N \times N$ weighted adjacency matrix $\widetilde{A}$
1: $A_{N \times N} \leftarrow J_{N \times N} - I_{N \times N}$ adjacency matrix of a complete graph
2: $\widetilde{A} \leftarrow b \cdot (A \circ D)$ weighted adjacency matrix
3: $S_{N \times N} \leftarrow$ Shortest path weight matrix of $\widetilde{A}$
4: $\hat{W}_{N \times N} \leftarrow$ Inverse link weight matrix of $\widetilde{A}$
5: **do**
6: $\quad \hat{\Omega}_{N \times N} \leftarrow$ Effective resistance matrix of $\hat{W}$
7: $\quad R \leftarrow \left(\hat{\Omega} - \hat{W}\right) \circ (D - S) \circ A$
8: $\quad (i, j) \leftarrow$ Indices of the maximum element in R
9: $\quad A \leftarrow A - e_i \cdot e_j^T - e_j \cdot e_i^T$
10: $\quad \widetilde{A} \leftarrow b \cdot (A \circ D)$
11: $\quad S_{N \times N} \leftarrow$ Shortest path weight matrix of $\widetilde{A}$
12: $\quad \hat{W}_{N \times N} \leftarrow$ Inverse link weight matrix of $\widetilde{A}$
13: **while** $(S \preccurlyeq D) \wedge (R_{ij} > 0)$
14: $A \leftarrow A + e_i \cdot e_j^T + e_j \cdot e_i^T$
15: $\widetilde{A} \leftarrow b \cdot (A \circ D)$
16: **return** $\widetilde{A}$

10. Link existence overrules the link weight. Equation (14) shows that if a link l_{ij} does not exist in graph G (i.e. $\tilde{a}_{ij} = 0$), than $\tilde{w}_{ij} = 0$, although $\frac{1}{\tilde{a}_{ij}} \to \infty$.

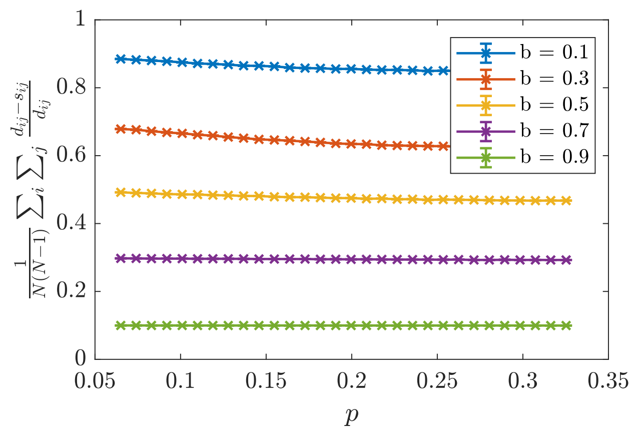
11. For any pair of connected nodes i and j we observe $s_{ij} = b \cdot d_{ij}$. In addition, for non-adjacent nodes m and n we reason $s_{mn} > b \cdot d_{mn}$, because D is a distance matrix. Combining these two observations, we conclude $S \leq b \cdot D$, which yields $||D - S|| < 1 - b$.

tree. Therefore, there are generally up to $\frac{N\cdot(N-1)}{2}-(N-1)$ iterations. The effective resistance and the shortest path weight between all node pairs are computed within each iteration. Within each iteration in our OLR, the effective resistance and the shortest path weight between any pair of nodes are computed. Both operations require computational complexity $O(N^3)$. In addition, we initialise OLR with a complete graph. The number of iterations in worst case scales as $O(N^2)$. Therefore,the overall complexity of our OLR is $O(N^5)$. Alternatively, DOR can streamline OLR's computational complexity. DOR ensures the retrieval of a graph with the minimum necessary links, accurately aligning the shortest path weight matrix S with the demand matrix D. Instead of initializing OLR with a complete graph, we employ DOR as the initial phase within OLR. Subsequently, we iteratively refine the graph until the shortest path weights fall within a predefined range, as dictated by the input parameter b. Consequently, the number of removed links within OLR reduces significantly, lowering its computational complexity to be $O(N^3L')$, where L' is the number of links in graph obtained by DOR.

Fig. 3 shows the differences between the number of links in the OLR graph H and the original graph G with different b and the norm $||D-S||=\frac{1}{N(N-1)}\sum_i\sum_j\frac{d_{ij}-s_{ij}}{d_{ij}}$ of the graph obtained by OLR with different input parameter b. For each simulation, we generate a 20-node Erdős–Rényi (ER) random graphs $G_p(20)$ and compute the corresponding shortest path weight matrix as the input demand matrix D, where p is the probability of connecting two nodes (link density). The link weights of the ER graph $G_p(20)$ are uniformly distributed in $(0,1)$. For each link density p, 1000 realizations are carried out. Fig. 3(a) illustrates that a smaller b generates a graph H with fewer links, while Fig. 3(b) shows that a smaller b corresponds to a large norm $||D-S||$.



(a) Number of additional links in obtained graph H



(b) Number of common links in G and H

Fig. 3. (a) Differences between number of links in the graph obtained by OLR and the original graph with different input parameter b. The x-axis denotes the link density p of the underlying 20-node ER graphs, while the y-axis is the difference between number of links in the graph H obtained by OLR and in the original graph G. (b) The norm $||D-S||$ of the graph obtained by OLR with different input parameter b. The x-axis denotes the link density p of the underlying 20-node ER graphs, while the y-axis is the norm $||D-S||$.

5 Performance Evaluation of DOR and OLR

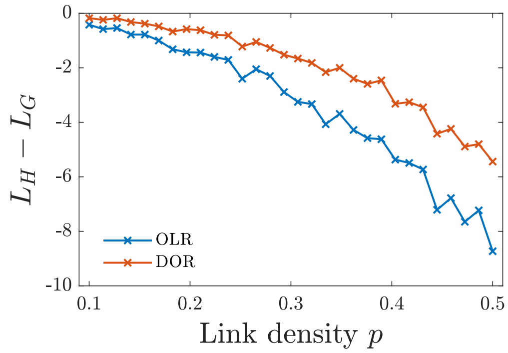
In this section, we evaluate the performance of DOR and OLR [12]. in random graphs and an empirical network. The performance of the DOR and OLR is assessed by three complementary criteria: (i) the number L_H-L_G of additional links in the resulting graph H, (ii) the number $\frac{1}{2L_H}\cdot u^T\cdot(A\circ A_H)\cdot u$ of common links in the original graph G and the resulting graph H and (iii) the norm $||D-S||=\frac{1}{N(N-1)}\sum_i\sum_j\frac{d_{ij}-s_{ij}}{d_{ij}}$ of the demand matrix D and the shortest path weight matrix S.

Fig. 4 illustrates the results of DOR (red line) and OLR (blue line) in ER graphs $G_p(N)$ with $N=10$ nodes and different link density p. We uniformly assign a random weight from $(0,1)$ to each link in G, thus defining the weighted adjacency matrix $\widetilde{A}$. For each generated ER graph, we provide the shortest path weight matrix of G as the input demand matrix D to the algorithm DOR and OLR. The input parameter of OLR $b=0.7$. We then obtain the resulting graph H, whose shortest path weight matrix is denoted as S. For each number N of nodes and different link density p, 100 simulation instances are executed and the average over 100 times of each criterion is computed.

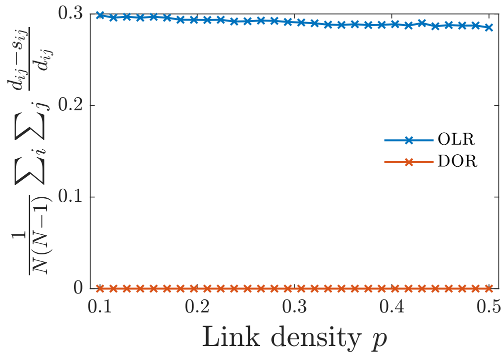
12. Matlab code is on https://github.com/qzhszl/IASPP.git

Fig. 4(a) depicts the difference in the number of links L_H-L_G between the obtained graph H and the original graph G. For a small link density p, the obtained graph H contains almost the same number of links L_H as that of the original graph L_G, while L_H-L_G decreases with the increment of link density p. As for the number of common links in the original graph G and the resulting graph H, our simulation (details are shown in Appendix G) shows that $\frac{1}{2L_H}\cdot u^T\cdot(A\circ A_H)\cdot u=1$ holds for both DOR and OLR with different link density p, which informs us that links of graph H obtained by both DOR and OLR belong to the original graph G. Fig. 4(b) illustrates the norm $||D-S||$, where DOR always returns an exact solution $||D-S||=0$ to the OIASPP. In contrast, for OLR, the norm $||D-S||$ is not zero but bounded by $1-b$.

A similar pattern in performance is visible for a different number of nodes N, as presented in Fig. 12 (for the case $N=20$) and in Fig. 13 (where the graph consists of $N=50$ nodes) in Appendix G. The feasibilities of DOR and OLR are also verified in Barabási–Albert (BA) networks [30] with $500, 1000$ and 10000 nodes, Watts–Strogatz (WS) small world network [31] with $100, 1000$ and 10000 nodes



(a) Number of additional links in obtained graph H



(b) Norm $||D - S||$

Fig. 4. Performance of the DOR and OLR on ER graphs with $N = 10$ nodes and different link density p. The input parameter $b = 0.7$.

and an empirical network USAir [32]. The details are shown in Appendix G.

In summary, the performance of DOR and OLR are stable with arbitrary demands on both small-size and large-size networks. Specifically, our simulation results verify that DOR provides a sparse graph that solves OIASPP exactly, while OLR exhibits a capacity to obtain a graph with fewer links compared with the DOR algorithm, at the cost of increased norm of $||D - S||$. The norm for DOR is always $||D - S|| = 0$, while for OLR $||D - S|| < 1 - b$, where $b \in [0, 1]$ is the input parameter.

6 Application

In this section, we discuss various IASPP applications and present a simulation example to validate the feasibility of our proposed DOR and OLR algorithms.

6.1 Application of IASPP

The IASPP methodology is useful in Wireless Sensor and Actuator Network (WSAN) [1]. Industrial WSAN (IWSAN) standards such as WirelessHART [33] have gained popularity in process automation, e.g., gas production, electric power generation and smelting plants. An IWSAN consists of a gateway, multiple access points and hundreds of thousands of field devices (i.e., sensors and actuators) that operate at low-power, forming a multi-hop wireless network, where the link weight w_{ij} between node i and node j denotes the latency bound that a link l_{ij} should provide. In a WSAN network, IASPP considers the end-to-end (E2E) latency as a demand matrix. The WSAN gateway

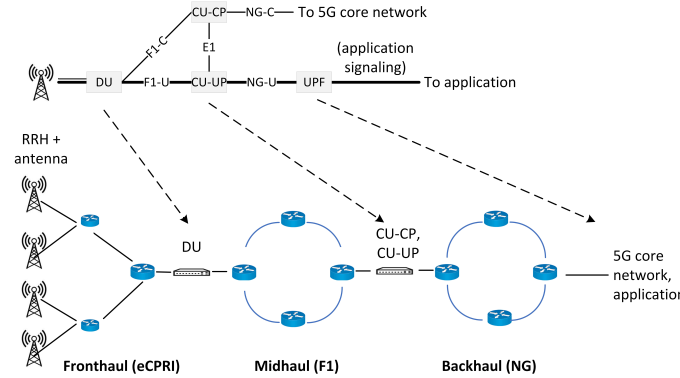


Fig. 5. A conceptual diagram of a RAN as found in the 5G mobile communications network.

collects network topology and flow demand information [34]. If there is topological change (e.g., node failure, new joining nodes) or change of the traffic pattern that makes current link weight configuration inappropriate[13], then the WSAN gateway can use DOR or OLR to (re-)computes the weighted adjacency matrix $\widetilde{A}$. The updated link weights will then be communicated with devices in the network. With the set of newly computed shortest paths, E2E latency of an arbitrary pair of nodes is guaranteed. A further step is to consider scheduling, power consumption and path redundancy into the problem.

Mobile communication radio access network [2] (RAN) is another application domain. Fig. 5 provides a conceptual diagram of a RAN as found in the 5G mobile communications network. The lower part of Fig. 5 depicts that the communication between the logical components [2] of the RAN (DU, CU-CP etc.) is formed by IP infrastructure [35]. Data transmission latency between the RAN logical components is bound by demands, i.e. maximum permissible E2E latency. With predetermined E2E latency demands, DOR and OLR can provide guidance in constructing a RAN network, such as installing base stations at different locations of a city.

Transportation networks constitute another potential application domain. For example, urban planners and customers may have demands on the commute time for each pair of bus or train stations. DOR can offer a transportation network such that the commute time between every two nodes (which denotes stations) exactly equals the prescribed demand and reaches a minimum number of links of all the networks with the same shortest path weight matrix. OLR can deal with more specific scenarios. Imagine urban planners have defined maximum allowable travel times as the demands for specific node pairs, accounting for variables like passenger density along these routes. The link weights represent the time needed when travelling between adjacent nodes. These maximum travel time constraints can span from 100% to about 200% of the calculated minimum travel time. OLR can shape the network into an ideal structure, while choosing a relatively small input parameter b value. This strategy seeks to mould the network's topology in a manner that caters to all essential routes while conforming to the stipulated upper travel time limits. By applying the

13. *Inappropriate* in this context means latency bound violation.

OLR algorithm, we can intelligently eliminate links, while preserving the network's overall connectivity and functionality. This process facilitates the creation of an optimised railway system that ensures both efficiency and adherence to travel time constraints. In the resultant graph generated by OLR, each link signifies the potential introduction of a direct line, further enhancing the network's efficiency and structure.

6.2 Simulation on E2E latency

In this section, we apply our IASPP methods to an E2E latency instance. Since the IASPP methods begin with a demand matrix which is also a distance matrix, the given demand matrix D is required to be modified so that we can imply our algorithm. If the E2E constraint of a node pair (i, j) is not specified, we assume that there is no constraint and that $d_{ij} = \infty$, which means there is no upper bound for the shortest path weight s_{ij} between node i and j. In many practical scenarios, not every pair of nodes necessarily has a demand. We first symmetrize the demand matrix (see explained in Section 2.1) following line $2-4$ of Algorithm 3. We then focus on the triangle inequality of a demand matrix. Consider the following example of a demand matrix:

$$D = \begin{bmatrix} 0 & 1 & \infty & \infty & 1 \\ 1 & 0 & 1 & 1 & \infty \\ \infty & 1 & 0 & 3 & \infty \\ \infty & 1 & 3 & 0 & 5 \\ 1 & \infty & \infty & 5 & 0 \end{bmatrix} \tag{15}$$

The demand $d_{34} > d_{32} + d_{24}$ is a typical case of the violation of the triangle inequality. However, the infinite $d_{ij} = \infty$ may lead to complicated cases. Aside from demands that violate the triangle inequality (e.g. d_{34}), there could be other demands that are unattainable. For instance, d_{45} does not breach the triangle inequality as $d_{45} < d_{14} + d_{15}$, $d_{45} < d_{24} + d_{25}$ and $d_{45} < d_{34} + d_{35}$. Nevertheless, d_{45}, d_{42}, d_{21} and d_{15} form a cycle structure and $d_{45} > d_{42} + d_{21} + d_{15}$. Consequently, s_{45} must be smaller than d_{45}, that is, d_{45} is not achievable.

To ensure all the demands are possible to achieve, we modify the demand matrix according to line $5 - 13$ of Algorithm 3. Our main idea is to calculate the shortest path weight matrix S of a graph whose weighted adjacency matrix equals the given demand matrix D, because the resulting demand matrix $D' = S$ reserves all the constraints in D except for the E2E demands not achievable. We can now transform an arbitrary non-negative demand matrix to a distance matrix and apply our DOR and OLR algorithms with the demand matrix D' as input.

We present an example in Fig. 6. Consider an E2E demand matrix:

$$D = \begin{bmatrix} 0 & 100 & 500 & \infty & \infty & 5000 & \infty & \infty & \infty & 100 \\ 100 & 0 & \infty & \infty & 20 & 500 & 20 & \infty & \infty & \infty \\ 500 & \infty & 0 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty & \infty & \infty & 50000 & \infty & \infty \\ \infty & 20 & \infty & \infty & 0 & 1000 & \infty & \infty & 20 & \infty \\ 5000 & 500 & \infty & \infty & 1000 & 0 & \infty & \infty & \infty & \infty \\ \infty & 20 & \infty & \infty & \infty & \infty & 0 & 500 & 100000 & 100 \\ \infty & \infty & \infty & 50000 & \infty & \infty & 500 & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & 20 & \infty & 100000 & \infty & 0 & \infty \\ 100 & \infty & \infty & \infty & \infty & \infty & 100 & \infty & \infty & 0 \end{bmatrix} \tag{16}$$

We first transform the given demand matrix D to D' following the method introduced by Algorithm 3. Our DOR

Algorithm 3 Demand Modification

Input: Demand matrix D whose unspecified demands are represented by ∞
Output: Modified demand matrix D'
1: $D' \leftarrow D$
2: **while** symmetry of D' is violated, i.e. $d_{ij} \neq d_{ji}$ **do**
3: $\quad d'_{ij} \leftarrow \min(d_{ij}, d_{ji})$, $d'_{ji} \leftarrow \min(d_{ij}, d_{ji})$
4: **end while**
5: **while** $d'_{ij} = \infty$ **do**
6: $\quad d'_{ij} \leftarrow 0$
7: **end while**
8: $G_{D'} \leftarrow$ Graph whose weighted adjacency matrix equals D'
9: $S \leftarrow$ Shortest path weight matrix of $G_{D'}$
10: **while** $s_{ij} = \infty$ **do**
11: $\quad s_{ij} \leftarrow$ Maximum finite element in S
12: **end while**
13: $D' \leftarrow S$
14: **return** D'

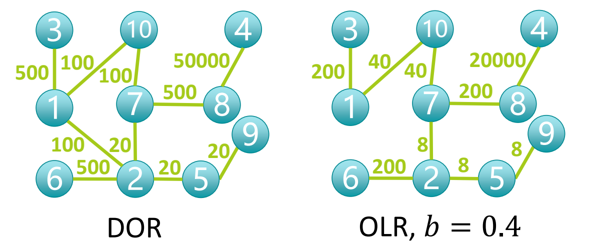


Fig. 6. Visualization of the graph H obtained by DOR and OLR, respectively. The input E2E demand matrix is (16).

and OLR algorithms were applied to the demand matrix D'. Each algorithm produced a graph H and the corresponding shortest path weight matrix S_1 for DOR and S_2 for OLR. These results are depicted in Fig. 6, Equation (17) and (18), respectively.

$$S_1 = \begin{bmatrix} 0 & 100 & 500 & 50620 & 120 & \mathbf{600} & 120 & 620 & 140 & 100 \\ 100 & 0 & 600 & 50520 & 20 & 500 & 20 & 520 & 40 & 120 \\ 500 & 600 & 0 & 51120 & 620 & 1100 & 620 & 1120 & 640 & 600 \\ 50620 & 50520 & 51120 & 0 & 50540 & 51020 & 50500 & 50000 & 50560 & 50600 \\ 120 & 20 & 620 & 50540 & 0 & \mathbf{520} & 40 & 540 & 20 & 140 \\ \mathbf{600} & 500 & 1100 & 51020 & \mathbf{520} & 0 & 520 & 1020 & 540 & 620 \\ 120 & 20 & 620 & 50500 & 40 & 520 & 0 & 500 & \mathbf{60} & 100 \\ 620 & 520 & 1120 & 50000 & 540 & 1020 & 500 & 0 & 560 & 600 \\ 140 & 40 & 640 & 50560 & 20 & 540 & \mathbf{60} & 560 & 0 & 160 \\ 100 & 120 & 600 & 50600 & 140 & 620 & 100 & 600 & 160 & 0 \end{bmatrix} \tag{17}$$

$$S_2 = \begin{bmatrix} 0 & 88 & 200 & 20280 & 96 & 288 & 80 & 280 & 104 & 40 \\ 88 & 0 & 288 & 20208 & 8 & 200 & 8 & 208 & 16 & 48 \\ 200 & 288 & 0 & 20480 & 296 & 488 & 280 & 480 & 304 & 240 \\ 20280 & 20208 & 20480 & 0 & 20216 & 20408 & 20200 & 20000 & 20224 & 20240 \\ 96 & 8 & 296 & 20216 & 0 & 208 & 16 & 216 & 8 & 56 \\ 288 & 200 & 488 & 20408 & 208 & 0 & 208 & 408 & 216 & 248 \\ 80 & 8 & 280 & 20200 & 16 & 208 & 0 & 200 & 24 & 40 \\ 280 & 208 & 480 & 20000 & 216 & 408 & 200 & 0 & 224 & 240 \\ 104 & 16 & 304 & 20224 & 8 & 216 & 24 & 224 & 0 & 64 \\ 40 & 48 & 240 & 20240 & 56 & 248 & 40 & 240 & 64 & 0 \end{bmatrix} \tag{18}$$

As demonstrated in (17), we highlight the shortest path weights which are different from the given specific E2E demands in (16). When we use DOR, all the shortest path weights are equal to the given E2E demands except for those that are not achievable. The OLR algorithm necessitates the input parameter b in addition to the demand matrix D', defining the allowed deviation of the norm of $||D - S||$ from 0. For the example illustrated in Fig. 6, we adopted $b = 0.4$. Lower values of b necessitate a reduced allocation of resources across the same set of links, culminating in diminished shortest path weights between all conceivable pairs of nodes. This outcome engenders sparser topologies due to the lowered link weights employed. Conversely, higher values of b impose greater link weights, which in turn lead to quicker breaches of the upper shortest path weight bounds provided in D during the iterative process. Therefore, a higher b value results in a higher-density topologies. Consequently, selecting the input parameter b represents a compromise between reducing the sparsity of the graph H topology and maximising the corresponding shortest path weights.

7 Conclusion

This work focuses on inverse all shortest path problem (IASPP), which is a novel problem with promising applications, such as network modelling and design, in transportation networks, wireless sensor and actuator networks, connected vehicle applications, smart factory networks, etc. We present the Descending Order Recovery (DOR) algorithm to solve the optimised inverse all shortest path problem (OIASPP) and prove that OIASPP is not NP-complete. The graph obtained by DOR does not have redundant links and reaches a minimum number of links and a minimum sum of the link weights among all OIASPP solutions given a demand matrix D. DOR can also be regarded as an effective method when solving network sparsification that preserves all shortest path weights. Additionally, we utilise the information captured by the effective resistance between node pairs and propose Omega-based Link Removal (OLR) algorithm that solves the OIASPP. Both DOR and OLR provide solutions to the OIASPP: the solution obtained by DOR has the shortest path weight matrix $S = D$, while OLR focuses on solving OIASPP by providing sparser graphs, at the cost of the norm $||D - S|| > 0$. The ideas of DOR and OLR are different: DOR focuses on the shortest paths and the shortest path weights in a graph, while OLR investigates the shortest path weights from the perspective of the effective resistance.

Acknowledgement

The authors would like to thank Massimo Achterberg for his valuable comments on earlier versions of this paper.

PVM is supported by the European Research Council under the European Union's Horizon 2020 research and innovation program (Grant Agreement 101019718).

References

[1] A. Nayak and I. Stojmenovic, *Wireless sensor and actuator networks: algorithms and protocols for scalable coordination and data communication*. Wiley Online Library, 2010.

[2] S. K. Singh, R. Singh, and B. Kumbhani, "The evolution of radio access network towards open-ran: Challenges and opportunities," in *2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 1–6, 2020.

[3] D. Burton and P. L. Toint, "On an instance of the inverse shortest paths problem," *Mathematical programming*, vol. 53, no. 1, pp. 45–61, 1992.

[4] D. Burton, W. Pulleyblank, and P. L. Toint, "The inverse shortest paths problem with upper bounds on shortest paths costs," in *Network Optimization*, (Berlin, Heidelberg), pp. 156–171, Springer Berlin Heidelberg, 1997.

[5] J. Zhou, F. Yang, and K. Wang, "An inverse shortest path problem on an uncertain graph," *Journal of Networks*, vol. 9, no. 9, p. 2353, 2014.

[6] D. Burton and P. L. Toint, "On the use of an inverse shortest paths algorithm for recovering linearly correlated costs," *Mathematical Programming*, vol. 63, no. 1, pp. 1–22, 1994.

[7] J. Zhang, Z. Ma, and C. Yang, "A column generation method for inverse shortest path problems," *Zeitschrift für Operations Research*, vol. 41, no. 3, pp. 347–358, 1995.

[8] S. Xu and J. Zhang, "An inverse problem of the weighted shortest path problem," *Japan journal of industrial and applied mathematics*, vol. 12, no. 1, pp. 47–59, 1995.

[9] S. P. Fekete, W. Hochstättler, S. Kromberg, and C. Moll, "The complexity of an inverse shortest path problem," in *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future*, vol. 49, pp. 113–127, 1999.

[10] P. Van Mieghem, "A tree realization of a distance matrix: the inverse shortest path problem with a demand matrix generated by a tree," *Delft University of Technology, Report20211012*, pp. 1–15, 2021.

[11] A. M. Mercier, S. V. Scarpino, and C. Moore, "Effective resistance for pandemics: Mobility network sparsification for high-fidelity epidemic simulation," *arXiv preprint arXiv:2111.02449*, 2021.

[12] P. Van Mieghem, *Graph Spectra for Complex Networks*. Cambridge University Press, second ed., 2023.

[13] P. Van Mieghem, *Performance analysis of complex networks and systems*. Cambridge, U.K.: Cambridge University Press, 2014.

[14] P. Van Mieghem, *Data Communications Networking*. Piet Van Mieghem, ISBN 978-94-91075-01-8, third ed., 2018.

[15] P. Van Mieghem and F. Kuipers, "On the complexity of qos routing," *Computer Communications*, vol. 26, no. 4, pp. 376–387, 2003. Quality of Future Internet Services.

[16] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[18] P. Van Mieghem and S. M. Magdalena, "Phase transition in the link weight structure of networks," *Phys. Rev. E*, vol. 72, p. 056138, Nov 2005.

[19] P. Van Mieghem and S. van Langen, "Influence of the link weight structure on the shortest path," *Phys. Rev. E*, vol. 71, p. 056113, May 2005.

[20] P. Van Mieghem and H. Wang, "The Observable Part of a Network," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 93–105, 2009.

[21] A. Ghosh, S. Boyd, and A. Saberi, "Minimizing effective resistance of a graph," *SIAM Review*, vol. 50, no. 1, pp. 37–66, 2008.

[22] W. Ellens, F. M. Spieksma, P. Van Mieghem, A. Jamakovic, and R. E. Kooij, "Effective graph resistance," *Linear algebra and its applications*, vol. 435, no. 10, pp. 2491–2506, 2011.

[23] P. Van Mieghem, K. Devriendt, and H. Cetinay, "Pseudoinverse of the Laplacian and best spreader node in a network," *Physical Review E*, vol. 96, no. 3, p. 032311, 2017.

[24] K. Devriendt, "Effective resistance is more than distance: Laplacians, simplices and the schur complement," *Linear Algebra and its Applications*, vol. 639, pp. 24–49, 2022.

[25] S. L. Hakimi and S. S. Yau, "Distance matrix of a graph and its realizability," *Quarterly of applied mathematics*, vol. 22, no. 4, pp. 305–317, 1965.

[26] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Mathematical programming*, vol. 27, no. 1, pp. 1–33, 1983.

[27] M. Fiedler, "Some characterizations of symmetric inverse M-matrices," *Linear algebra and its applications*, vol. 275-276, pp. 179–187, 1998.

[28] M. Fiedler, *Matrices and graphs in geometry*. Cambridge, U.K.: Cambridge University Press, 2011.

[29] T. Simas, R. B. Correia, and L. M. Rocha, "The distance backbone of complex networks," *Journal of Complex Networks*, vol. 9, 10 2021.

[30] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 74, pp. 47–97, Jan 2002.

[31] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

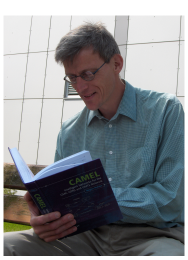
[32] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015.

[33] "Industrial communication networks-wireless communication network and communication profiles-WirelessHART™," *IEC standard 62591: Geneve, Switzerland*, p. 944, 2010.
[34] J. Schönwälder, M. Björklund, and P. Shafer, "Network configuration management using netconf and yang," *IEEE communications magazine*, vol. 48, no. 9, pp. 166–173, 2010.
[35] S. Rommel, R. Thiago Raddo, and I. T. Monroy, "The fronthaul infrastructure of 5g mobile networks," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pp. 1–6, 2018.

Zhihao Qiu is pursuing his PhD degree since October 2021 at Delft University of Technology, The Netherlands. He obtained the M.Sc. degree in Information Physics and the B.Sc. degree in Electronic Information Science and Technology at the University of Electronic Science and Technology of China, Chengdu, China. His main research interests include network science, shortest path problem, vital node identification and science of science.

Ivan Jokić is pursuing his PhD degree since February 2019 at Delft University of Technology, The Netherlands. He obtained the M.Sc. degree in Control Theory at the University of Montenegro, Podgorica, Montenegro, in 2018. In 2015, he obtained a B.Sc. degree in Energetics and Control Theory at University of Montenegro, Podgorica, Montenegro. His main research interests include graph theory, network dynamics, systems theory and networked systems identification.

Siyu Tang received her M.Sc. and Ph.D. degree in Electrical Engineering from Delft University of Technology, The Netherlands, in 2006 and 2010 respectively. Since then, she was with Bell Labs, Alcatel-Lucent (later merged with Nokia), Antwerp, Belgium, working on novel algorithms and network protocols for ultra-low latency networks. From 2017, she joined Huawei Munich Research Center, Germany, as a senior principal researcher. She leads several research activities in the field of Telecommunication networks (e.g., future Internet architecture and protocols) and Industrial Communication Networks (e.g., time sensitive networks, DetNet).



Rogier Noldus is principal solution architect at Ericsson Telecommunicatie, Rijen, Netherlands. His main line of work is (mobile) communication network architecture and services. He is the (co-) author of the books "CAMEL: Intelligent Networks for the GSM, GPRS and UMTS Network" and "IMS Application Developer's Handbook". He is in addition part-time lecturing at the TU Delft (Mobile network architecture), where he is also pursuing (part-time) a PhD in the area of network topology optimization. Rogier holds a BSc (electrical engineering) from the Hogeschool Utrecht, Netherlands, and an MSc (telecommunications) from the University of the Witwatersrand, Johannesburg, South Africa.

Piet Van Mieghem Piet Van Mieghem is professor at the Delft University of Technology and chairman of the section Network Architectures and Services (NAS) since 1998. He is the author of four books: Performance Analysis of Communications Networks and Systems, Data Communications Networking, Graph Spectra for Complex Networks and Performance Analysis of Complex Networks and Systems. He is a board member of the Netherlands Platform of Complex Systems, a steering committee member of the Dutch Network Science Society , an external faculty member at the Institute for Advanced Study (IAS) of the University of Amsterdam and an IEEE Fellow. He was awarded an Advanced ERC grant 2020 for ViSiON, Virus Spread in Networks. Professor Van Mieghem received a Master degree and a Ph.D. degree in Electrical Engineering from the K.U.Leuven (Belgium) in 1987 and 1991, respectively. Before joining Delft, he worked at the Interuniversity Micro Electronic Center (IMEC) from 1987 to 1991. During 1993 to 1998, he was a member of the Alcatel Corporate Research Center in Antwerp. He was a visiting scientist at MIT (1992-1993) and a visiting professor at UCLA (2005), at Cornell University (2009), at Stanford University (2015) and at Princeton University (2022). Currently, he serves on the editorial board of the OUP Journal of Complex Networks. He was member of the editorial board of Computer Networks (2005-2006), the IEEE/ACM Transactions on Networking (2008- 2012), the Journal of Discrete Mathematics (2012-2014) and Computer Communications (2012-2015).

A Nomenclature

TABLE 1
Nomenclature

Symbol	Definition
G	Graph
$\mathcal{N}$	Set of nodes
N	Number of nodes
$\mathcal{L}$	Set of links
L	Number of links
$\mathcal{P}_{ij}$	Path from node i to node j
$\mathcal{P}^*_{ij}$	Shortest path from node i to node j
D	Demand matrix
G_D	Complete graph whose weighted adjacency matrix equals the demand matrix D
A	Adjacency matrix
W	Link weight matrix
$\widetilde{A}$	Weighted adjacency matrix
$\widetilde{A}_F$	Weighted adjacency matrix of a flow network
Ω	Effective resistance matrix
r_l	Link l resistance
R_G	Effective graph resistance
S	Shortest path weight matrix
Q	Laplacian matrix
$Q^\dagger$	Pseudoinverse Laplacian matrix
$\widetilde{\Delta}$	Weighted degree matrix
$\widetilde{\Delta}_F$	Weighted degree matrix of a flow network
d_i	Degree of node i
u	All-one vector
J	All-one matrix
I	Identity matrix
e_i	$N \times 1$ basic vector has only one non-zero element $(e_i)_i = 1$

B Flow analogue method for general graphs

As demonstrated in Section 2.2, Van Mieghem [1] proposed the "flow analogue method" to recover the weighted adjacency matrix $\widetilde{A}$ from a demand matrix D, which is also a shortest path weight matrix S, by exploiting the analogy between flow and path networks, when the given shortest path weight matrix S derived from a tree graph. The crux of the "flow analogue method" lies in the strategic employment of the equivalence existing between the shortest path matrix S and the effective resistance matrix Ω. Since the equality between the shortest path weight matrix S and the effective resistance matrix Ω only holds for the tree graph, the given demand matrix must be transformed into an effective resistance matrix first.

Unfortunately, it remains an open question how to construct an effective resistance matrix whose corresponding weighted adjacency matrix is non-negative. In Appendix C, we try to extend the flow analogue method utilizing perturbation theory. We first obtain a perturbed effective resistance matrix of a tree graph by adding an effective resistance matrix of a different tree graph. The resulting perturbed effective resistance matrix is then imported into (8) and we can obtain a weighted Laplacian and the corresponding weighted adjacency matrix $\widetilde{A}$. Appendix C demonstrates that even an extremely tiny perturbation of an effective resistance matrix of a tree graph may lead to negative off-diagonal elements in the obtained weighted adjacency matrix $\widetilde{A}$ computed with (12). Hence, the sum of two effective resistance matrices is not necessarily an effective resistance matrix.

The difficulty of constructing an effective resistance matrix with a non-negative corresponding weighted adjacency matrix can also be explained from the perspective of the inverse simplex [2] of a graph. The details of simplex and inverse simplex are provided in Appendix D. Any undirected, weighted graph G can be uniquely represented by a simplex $\mathcal{V}$ or an inverse simplex $\mathcal{V}^+$ in the $N-1$ dimensional Euclidean space [2]. The reverse also holds: Every simplex with non-obtuse angles, i.e. smaller than or equal to 90 degrees, between all pairs of facets is the inverse simplex $\mathcal{V}^+$ of a connected, undirected graph with positive link weights [2]. Fiedler [2], [3] demonstrated that the angle ϕ^+_{ij} between two facets $\mathcal{F}^+_{\bar{i}}$ and $\mathcal{F}^+_{\bar{j}}$ in an inverse simplex $\mathcal{V}^+$ is related to the graph G by

$$\cos(\phi^+_{ij}) = -\frac{\widetilde{q}_{ij}}{\sqrt{\widetilde{q}_{ii}\widetilde{q}_{jj}}} \tag{20}$$

where $\widetilde{q}_{ii}$ and $\widetilde{q}_{ij}$ are the diagonal and off-diagonal elements of the weighted Laplacian matrix $\widetilde{Q}$. As shown in (20), if there is no link between node i and j, then $\tilde{q}_{ij} = 0$ and the dihedral angle ϕ^+_{ij} exactly equals 90 degrees. For an inverse simplex of tree graph, there are $\frac{N(N-1)}{2} - (N-1)$ 90-degree dihedral angles. Moreover, a positive off-diagonal element $\widetilde{q}_{ij}$, which leads to a negative link weight, corresponds to an obtuse dihedral angle with more than 90 degrees. When we perturb the effective resistance matrix of a tree, the corresponding inverse simplex has a high probability of having obtuse dihedral angles because each 90-degree angle may become obtuse even with a tiny perturbation. This explains why a tiny perturbation on an effective resistance matrix of a tree graph may lead to negative off-diagonal elements in the obtained weighted adjacency matrix $\widetilde{A}$.

Fiedler [3] proposed that the effective resistance matrix Ω of a weighted graph G equals the squared Euclidean distance matrix D_E, for which $d_E(ij) = ||d_E(i) - d_E(j)||^2$ for the vertices coordinates $\{v_1, v_2, \ldots, v_N\}$ of the corresponding inverse simplex $\mathcal{V}^+$ in the Euclidean space. Constructing an effective resistance matrix whose corresponding weighted adjacency matrix is non-negative can thus be regarded as a problem in a graph: "Obtain the criterion for the edges in a simplex that does not have obtuse angles between arbitrary two facets", which is complicated.

C First-order perturbation

Given two effective resistance matrices Ω_1 and Ω_2 that are derived from tree graphs, we deduce the effect of perturbing the effective resistance matrix Ω_1 by $\varepsilon\Omega_2$, where ε is an arbitrary positive real number. By scaling Ω with ε, the relations $2\sigma^2 = \frac{1}{u^T\Omega^{-1}u}$ and the vector $p = \frac{1}{u^T\Omega^{-1}u}\Omega^{-1}u$ indicate that only σ^2 is multiplied by ε, but not the vector p.

After combining

$$\Omega^{-1} = \frac{1}{2\sigma^2}pp^T - \frac{1}{2}\widetilde{Q} \tag{21}$$

in [1] with $\Omega = \Omega_1 + \varepsilon\Omega_2$, we find

$$\begin{aligned}\Omega^{-1} &= (\Omega_1 + \varepsilon\Omega_2)^{-1} = \left(\Omega_1\left(I + \varepsilon\Omega_1^{-1}\Omega_2\right)\right)^{-1} \\ &= \left(I + \varepsilon\Omega_1^{-1}\Omega_2\right)^{-1}\Omega_1^{-1}\end{aligned}$$

where $(AB)^{-1} = B^{-1}A^{-1}$ (see e.g. [4, p. 93]) is used. Invoking $(I+\varepsilon R)^{-1} = \sum_{k=0}^{\infty}(-1)^k \varepsilon^k R^k$ for sufficiently small $\varepsilon < \frac{1}{\lambda_{\max}(R)}$,

$$\begin{aligned}\Omega^{-1} &= \left(I+\sum_{k=1}^{\infty}(-1)^k \varepsilon^k \left(\Omega_1^{-1}\Omega_2\right)^k\right)\Omega_1^{-1}\\ &= \Omega_1^{-1}+\sum_{k=1}^{\infty}(-1)^k \varepsilon^k \left(\Omega_1^{-1}\Omega_2\right)^k \Omega_1^{-1}\\ &= \Omega_1^{-1}-\varepsilon\left(\Omega_1^{-1}\Omega_2\right)\Omega_1^{-1}+\varepsilon^2\left(\Omega_1^{-1}\Omega_2\right)^2\Omega_1^{-1}+O\left(\varepsilon^3\right)\end{aligned}$$

and ignoring higher order terms $O\left(\varepsilon^2\right)$ yields

$$\Omega^{-1} \approx \Omega_1^{-1} - \varepsilon\Omega_1^{-1}\Omega_2\Omega_1^{-1}$$

Then

$$\begin{aligned}\frac{1}{2\sigma^2} &= u^T\Omega^{-1}u \approx u^T\Omega_1^{-1}u^T - \varepsilon u^T\Omega_1^{-1}\Omega_2\Omega_1^{-1}u\\ &= \frac{1}{2\sigma_1^2}\left(1-\frac{\varepsilon}{2\sigma_1^2}p_1^T\Omega_2 p_1\right)\end{aligned}$$

and

$$\begin{aligned}p &= 2\sigma^2\Omega^{-1}u \approx 2\sigma^2\Omega_1^{-1}u - \varepsilon 2\sigma^2\Omega_1^{-1}\Omega_2\Omega_1^{-1}u\\ &= \frac{\sigma^2}{\sigma_1^2}\left(p_1-\varepsilon\Omega_1^{-1}\Omega_2 p_1\right)\\ &\approx \frac{1}{1-\frac{\varepsilon}{2\sigma_1^2}p_1^T\Omega_2 p_1}\left(p_1-\varepsilon\Omega_1^{-1}\Omega_2 p_1\right)\\ &\approx \left(1+\frac{\varepsilon}{2\sigma_1^2}p_1^T\Omega_2 p_1\right)\left(p_1-\varepsilon\Omega_1^{-1}\Omega_2 p_1\right)\\ &= p_1+\varepsilon\left(\frac{p_1^T\Omega_2 p_1}{2\sigma_1^2}-\Omega_1^{-1}\Omega_2\right)p_1\end{aligned}$$

Finally, to first order in ε, we obtain (22) with $\Omega^{-1} = \frac{1}{2\sigma^2}p.p^T - \frac{1}{2}\widetilde{Q}$. Then (23) relates the perturbed Laplacian with the effective resistance matrices Ω_1 and Ω_2, which is fairly complicated.

Consider two trees T_1 and T_2 with the same number of nodes N. Since T_1 and T_2 are trees, it holds that the effective resistance matrix equals to the shortest path weight matrix, i.e. $S_1 = \Omega_1$ and $S_2 = \Omega_2$. Therefore, Fiedler's block matrix relation (8) applies to these effective resistance matrices and shortest path matrices. We assume that T_2 is a small perturbation on the original graph T_1, which is reflected by the small perturbation strength ε. Then the total effective resistance matrix $\Omega = \Omega_1 + \varepsilon\Omega_2$. From (23), we derive the weighted Laplacian $\widetilde{Q}$ corresponding to Ω, which equals in terms of Ω_1 and Ω_2 as

$$\widetilde{Q} = \widetilde{Q}_1 + \varepsilon\Delta\widetilde{Q} + O\left(\varepsilon^2\right) \tag{24}$$

where the first-order term in the perturbation ε is

$$\begin{aligned}\Delta\widetilde{Q} =& \frac{p_1^T\Omega_2 p_1}{2\sigma_1^4}p_1p_1^T - \frac{1}{\sigma_1^2}\left(\Omega_1^{-1}\Omega_2 p_1p_1^T + p_1p_1^T\Omega_2^T\left(\Omega_1^{-1}\right)^T\right)\\ &+ 2\Omega_1^{-1}\Omega_2\Omega_1^{-1}\end{aligned} \tag{25}$$

where σ_1 and p_1 are calculated by (9) and (10). Equation in (24) is accurate if ε is sufficiently small. If we neglect the higher-order terms $O\left(\varepsilon^2\right)$, a corresponding element in (24) is

$$\widetilde{q}_{ij} = (\widetilde{q}_1)_{ij} + \varepsilon\Delta\widetilde{q}_{ij} \tag{26}$$

For any tree T_1 with $N > 2$, there exist zero values in the off-diagonal elements of the weighted Laplacian matrix $\tilde{Q}_1$, because these zero values correspond to non-existing links. Consider such an element (i.e. $\widetilde{q}_{ij} = 0$), such that (26) simplifies to $\widetilde{q}_{ij} = \varepsilon\Delta\widetilde{q}_{ij}$. Since $\varepsilon > 0$, the sign of $\Delta\widetilde{q}_{ij}$ will determine the sign of $\widetilde{q}_{ij}$. Equation (25) tells us that $\Delta\widetilde{q}_{ij}$ is determined by the topology and the link weight structure of tree T_1 and tree T_2, which means that only the tree T_1 and T_2 determine the sign of $\widetilde{q}_{ij}$, irrespective of the perturbation strength ε. However, the off-diagonal elements $\widetilde{q}_{ij}$ in a weighted Laplacian $\widetilde{Q} = \widetilde{\Delta} - \widetilde{A}$ should be non-positive, because all the off-diagonal elements in a weighted adjacency matrix $\widetilde{A}$ are non-negative. The appearance of a positive off-diagonal $\widetilde{q}_{ij}$ or, equivalently, a negative off-diagonal $\widetilde{a}_{ij}$ indicates that the addition of an extremely small, but non-zero perturbation in $\Omega = \Omega_1 + \varepsilon\Omega_2$ does *not* create an effective resistance matrix Ω.

D Simplex and inverse simplex

Besides the adjacency matrix A, the Laplacian matrix Q and the effective resistance matrix Ω, which represent a graph in the topology domain, an undirected graph can be represented in the geometric domain by its corresponding simplex or inverse simplex [2]. Any undirected, weighted graph G can be uniquely represented by a simplex $\mathcal{V}$ or an inverse simplex $\mathcal{V}^+$ in the $N-1$ dimensional Euclidean space [2]. Both simplex $\mathcal{V}$ and inverse simplex $\mathcal{V}^+$ are geometric objects that generalizes triangles and tetrahedrons to any dimension [2]. For example, $0-$simplex, $1-$simplex, $2-$simplex and $3-$simplex respectively correspond to a point, a line segment, a triangle and a tetrahedron, as shown in Fig. 10. The vertex matrix $V = [v_1, v_2, \ldots, v_N]$, where v_i is a $N-1$ dimensional vector including the coordinates of the vertex i in an simplex $\mathcal{V}^+$, can be obtained from $\widetilde{Q} = V^TV$. In an inverse simplex $\mathcal{V}^+$, the vertex matrix $V^\dagger = [v_1^+, v_2^+, \ldots, v_N^+]$, where v_i^+ is a $N-1$ dimensional vector including the coordinates of the vertex i, can be calculated by $\widetilde{Q}^\dagger = V^{\dagger T}V^\dagger$, where $\widetilde{Q}^\dagger$ is the weighted pseudoinverse Laplacian matrix of G. The surface or boundary of a simplex $\mathcal{V}$ (or an inverse simplex $\mathcal{V}^+$) is called a face, which is a lower dimensional simplex [2]. A face in a simplex (or an inverse simplex) is denoted by $\mathcal{F}_{\mathcal{E}}$ (or $\mathcal{F}_{\mathcal{E}}^+$), where vector $\mathcal{E}$ is a subset of vertex indices of a simplex $\mathcal{V}$ (or an inverse simplex $\mathcal{V}^+$). If we use $\bar{\mathcal{E}}$ as the complementary set of vertex indices, $\mathcal{F}_{\mathcal{E}}$ and $\mathcal{F}_{\bar{\mathcal{E}}}$ are then called complementary faces. For example, in a 3-dimensional simplex (known as a tetrahedron), a 0-dimensional face corresponds to a vertex, while its complementary face is a triangle (2-dimensional face). Furthermore, a $N-2$ dimensional face is called a facet and the dihedral angles between two facets of a simplex $\mathcal{V}$ (or an inverse simplex $\mathcal{V}^+$) are denoted by ϕ (or ϕ^+).

E Three NP-complete ISPP

E.1 A general inverse shortest path problem studied by Fekete et al.

In 1999, Fekete et al. [5] consider a general ISPP ($\text{ISPP}_{\text{Fekete}}$), where only the shortest path weight between pairs of nodes are given, but not the paths achieving them.

$$
\begin{aligned}
\widetilde{Q} &\approx \frac{1}{\sigma^2} p p^T - 2\Omega_1^{-1} + 2\varepsilon\Omega_1^{-1}\Omega_2\Omega_1^{-1} \\
&\approx \frac{1}{\sigma_1^2}\left(1 - \frac{\varepsilon}{2\sigma_1^2} p_1^T \Omega_2 p_1\right)\left(p_1 + \varepsilon\left(\frac{p_1^T\Omega_2 p_1}{2\sigma_1^2} - \Omega_1^{-1}\Omega_2\right)p_1\right)\left(p_1 + \varepsilon\left(\frac{p_1^T\Omega_2 p_1}{2\sigma_1^2} - \Omega_1^{-1}\Omega_2\right)p_1\right)^T \\
&\quad - 2\Omega_1^{-1} + 2\varepsilon\Omega_1^{-1}\Omega_2\Omega_1^{-1} \qquad (22)\\
&= \frac{1}{\sigma_1^2}\left(1 - \frac{\varepsilon}{2\sigma_1^2} p_1^T \Omega_2 p_1\right) p_1 p_1^T + \frac{\varepsilon}{\sigma_1^2}\left\{\frac{p_1^T\Omega_2 p_1}{\sigma_1^2} p_1 p_1^T - \Omega_1^{-1}\Omega_2 p_1 p_1^T - p_1 p_1^T \Omega_2^T \left(\Omega_1^{-1}\right)^T\right\} - 2\Omega_1^{-1} + 2\varepsilon\Omega_1^{-1}\Omega_2\Omega_1^{-1} \\
&= \frac{1}{\sigma_1^2} p_1 p_1^T - 2\Omega_1^{-1} + \frac{\varepsilon}{\sigma_1^2}\left\{\frac{p_1^T\Omega_2 p_1}{2\sigma_1^2} p_1 p_1^T - \Omega_1^{-1}\Omega_2 p_1 p_1^T - p_1 p_1^T \Omega_2^T \left(\Omega_1^{-1}\right)^T\right\} + 2\varepsilon\Omega_1^{-1}\Omega_2\Omega_1^{-1}
\end{aligned}
$$

$$
\widetilde{Q} = \widetilde{Q}_1 + \varepsilon\left\{\frac{p_1^T\Omega_2 p_1}{2\sigma_1^4} p_1 p_1^T - \frac{1}{\sigma_1^2}\left(\Omega_1^{-1}\Omega_2 p_1 p_1^T + p_1 p_1^T \Omega_2^T \left(\Omega_1^{-1}\right)^T\right) + 2\Omega_1^{-1}\Omega_2\Omega_1^{-1}\right\} + O\left(\varepsilon^2\right) \qquad (23)
$$

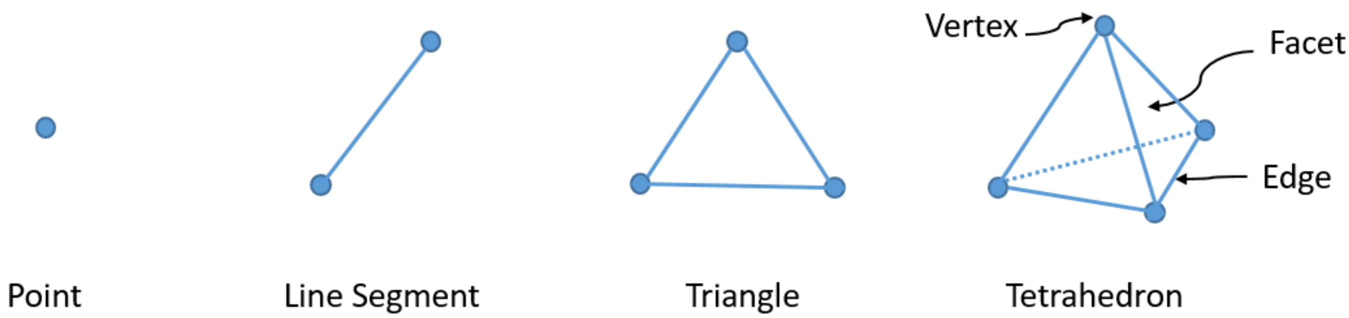


Fig. 10. Examples of low-dimensional simplices

A general inverse shortest path problem studied by Fekete et al. ($\text{ISPP}_{\text{Fekete}}$**)**: Given a N-node graph G with adjacency matrix A and a $N \times N$ symmetric and non-negative demand matrix D. Determine the link weight matrix W and the weighted adjacency matrix $\widetilde{A}$ such that the corresponding shortest path weight matrix S equals D.

In $\text{ISPP}_{\text{Fekete}}$, the demand matrix D does not necessarily include all the shortest path weights of the pairs of nodes in graph G. Fekete et al. [5] prove that $\text{ISPP}_{\text{Fekete}}$ is NP-complete by reducing $\text{ISPP}_{\text{Fekete}}$ to a vertex-disjoint paths problem, which is NP-complete. Specifically, $\text{ISPP}_{\text{Fekete}}$ is polynomial solvable if the distance graph $G_d(V, E_d, w_d)$ is a star (in which all the links are incident to a node) or the union of complete star (which is a star include all the nodes in the graph). The problem becomes NP-complete if slightly more complicated distance graphs $G_d(V, E_d, w_d)$ are considered.

E.2 Forth path cut problem

B. A. Miller et al. [6] investigated an ISPP aiming to obtain a graph, such that a given path is the shortest path, by removing links of the given graph. This problem is referred as Forth Path Cut problem (FPCP).

Forth path cut problem: Given a weighted graph G with link weight matrix W and a path $\mathcal{P}_{ij}$ for node i and j, determine a new weighted graph G' by removing links, such that $\mathcal{P}$ is the shortest paths in the graph and the sum of the weights of removed links does not exceed a given limitation b. B. A. Miller et al. [6] proved that the Force path cut problem is an NP-complete problem by reducing FPCP to 3-Terminal Cut problem, which is NP-complete. Given a graph G with weights matrix W and three terminal nodes, the 3-Terminal cut problem asks for removing a set of links $\mathcal{L}_{\text{R}} \in \mathcal{L}$ such that the terminals are disconnected (there is no path connecting any two terminals) in the resulting graph G' and the sum of weights of removed links $l \in \mathcal{L}_{\text{R}}$ does not exceed a given limitation b. The success condition of Force Path Cut is that all paths from nodes i to j aside from $\mathcal{P}$ must be strictly longer than $\mathcal{P}$, which is an example of the Weighted Set Cover problem. In Force Path Cut, the elements of the universe to cover are the paths and the sets represent links: each link corresponds to a set containing all paths from s to t on which it lies. Including this set in the cover implies removing the edge, thus covering the elements (i.e., cutting the paths). While the weighted cover is computationally intractable, there are established approximation algorithms. B. A. Miller et al. [6] then proposed an algorithm called path attack, which uses a natural oracle to generate only those constraints needed to execute the approximation algorithm.

E.3 ISPP with upper bound

Burton et al. [7] posed an ISPP with upper bound constraints, i.e. the weight of the shortest path $s_{ij} = w(\mathcal{P}^*_{ij}) \leq d_{ij}$, where d_{ij} is the element of the demand matrix D

between nodes i and j, which is called "upper bound". The problem is described as

The inverse shortest path problem with upper bound constraints($\mathrm{ISPP}_{\mathrm{upperbound}}$): Given a graph G with link weight matrix W and an $N \times N$ symmetric demand matrix D with zero diagonal elements, but positive off-diagonal elements. Determine a new link weight matrix W', such that the corresponding shortest path weight matrix S obeys $S \preccurlyeq D$ and minimizes a norm $||W' - W||$.

In this case, the "upper bound" d_{ij} are allowed to be infinite, i.e. the constraint of "upper bond" is not necessary for all the shortest path weights. Burton et al. [7] choose l_2 norm and consider $\mathrm{ISPP}_{\mathrm{upperbound}}$ as a least squares problem

$$\min_{w'_l \in \mathcal{L}'} \frac{1}{2} \sum_{l=1}^{L} (w'_l - w_l)^2 \tag{27}$$

where w_l and w'_l denote the weight of link $l \in \mathcal{L}$ and $l \in \mathcal{L}'$ respectively, subject to

$$w_l \geq 0, l \in \mathcal{L} \tag{28}$$

and the bound constraints on the shortest paths,

$$\sum_{l \in \mathcal{P}^*_{ij}} w_l \leq d_{ij} \tag{29}$$

Equation (29) only consider the shortest path weight and the definition of the path $\mathcal{P}^*_{ij}$ is implicit. For example, as the link weight matrix W is modified, the nodes and links belonging to the shortest path $\mathcal{P}^*_{ij}$ between nodes i and nodes j may vary. Burton et al. [7] illustrated the non-convex nature of the "upper bound" constraints and the least squares problem describing $\mathrm{ISPP}_{\mathrm{upperbound}}$ is proved to be non-convex. The quadratic programming algorithm introduced in [8] thus cannot be utilized directly to solve $\mathrm{ISPP}_{\mathrm{upperbound}}$.

Burton et al. [7] then proved that finding a global solution of $\mathrm{ISPP}_{\mathrm{upperbound}}$ is NP-complete through a decision problem ISP: Given an $\mathrm{ISPP}_{\mathrm{upperbound}}$ with a bound k, does there exists a solution with objective value at most k? Burton et al. demonstrate that (a) the decision question ISP is in NP. (b) a transformation from 3-SAT problem [9] to ISP can be constructed (c) the transformation is proved to be polynomial. The decision question ISP is thus NP-complete and $\mathrm{ISPP}_{\mathrm{upperbound}}$ is NP-hard.

Since the $\mathrm{ISPP}_{\mathrm{upperbound}}$ is non-convex and NP-hard, Burton et al. [7] proposed a local optimum algorithm to solve the shortest path problem. A feasible starting point is computed first. Then at each iteration, with a modified weight matrix W', the explicit definition of the shortest path constraints(29) is revised and the resulting convex problem is solved by the algorithm introduced in [7], which leads a new modified weight matrix W'. The calculation stops when no further progress can be obtained. The algorithm terminates in a finite number of iterations.

F Pseudocode of DOR initialised with a tree

In this section, we supplement the pseudocode of DOR algorithm initialised with a tree, which is shown in Algorithm 4.

Algorithm 4 Descending Order Recovery (DOR) initialised with a tree graph

Require: $N \times N$ demand matrix $D = S$: a shortest path weight matrix of a graph G
Ensure: $N \times N$ weighted adjacency matrix $\widetilde{A}$
$\widetilde{A}_D \leftarrow D$
$G_D \leftarrow$ Complete graph whose weighted adjacency matrix is $\widetilde{A}_D$
$T_D \leftarrow$ Minimum spanning tree of G_D
$\widetilde{A} \leftarrow$ Weighted adjacency matrix of T_D
$S' \leftarrow$ Shortest path weight matrix of T_D
while $C \leftarrow D - S'$ has at least one negative element, i.e. $(c_{ab})_{\max} > 0$ **do**
    $(i, j) \leftarrow$ Indices of the maximum element in C
    $\widetilde{a}_{ij} \leftarrow d_{ij}$, $\widetilde{a}_{ji} \leftarrow d_{ij}$
    $G_A \leftarrow$ Graph whose weighted adjacency matrix is $\widetilde{A}$
    $S' \leftarrow$ Shortest path weight matrix of G_A
end while
return $\widetilde{A}$

G Performance of the DOR and OLR algorithm

In this section, we supplement the performance of DOR and OLR in ER random network $G_p(N)$, Barabási–Albert (BA) network [10], Watts–Strogatz(WS) small world network [11] and an empirical network USAir [12]. The performance of the DOR and OLR is assessed by three complementary criteria: (i) the number $L_H - L_G$ of additional links in the resulting graph H, (ii) the number $L_C = \frac{1}{2L_H} \cdot u^T \cdot (A \circ A_H) \cdot u$ of common links in the original graph G and the resulting graph H and (iii) the norm $||D - S|| = \frac{1}{N(N-1)} \sum_i \sum_j \frac{d_{ij} - s_{ij}}{d_{ij}}$ of the demand matrix D and the shortest path weight matrix S.

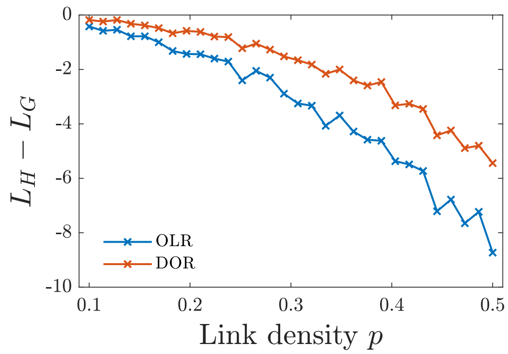
Fig. 11 (for the case $N = 10$), Fig. 12 (for the case $N = 20$) and Fig. 13 (where the graph consists of $N = 50$ nodes) illustrate the results for ER graphs. We uniformly assign a random weight from $(0, 1)$ to each link in G, thus defining the $N \times N$ weighted adjacency matrix $\widetilde{A}$. For each generated ER graph, we provide the $N \times N$ shortest path weight matrix of G as the input demand matrix D to the algorithm DOR and OLR. The input parameter of OLR $b = 0.7$. We then obtain the resulting graph H, whose $N \times N$ shortest path weight matrix is denoted as S. For each number N of nodes and different link density p, 100 simulation instances are executed and the average over 100 times of each criterion is computed.

TABLE 2
Performance of the DOR

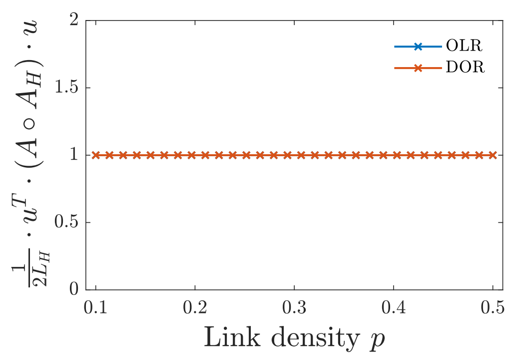
Network	N	L_G	$L_H - L_G$	L_C	$\|\|D-S\|\|$
BA1	500	994	-72	1	0
BA2	500	1979	-750	1	0
BA3	1000	1993	-76	1	0
BA4	10000	19989	0	1	0
WS1	100	200	-9	1	0
WS2	100	300	-65	1	0
WS3	1000	3000	-244	1	0
WS4	10000	30000	0	1	0
USAir	332	2126	-279	1	0

TABLE 3
Performance of the OLR

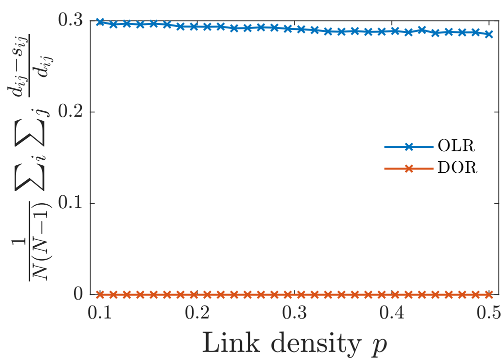
Network	N	L_G	$L_H - L_G$	L_C	$\|\|D-S\|\|$
BA1	500	994	-181	1	0.6736
BA2	500	1979	-853	1	0.6938
BA3	1000	1993	-140	1	0.6961
BA4	10000	19989	-36	1	0.6939
WS1	100	200	-39	1	0.6865
WS2	100	300	-157	1	0.6642
WS3	1000	3000	-995	1	0.6850
WS4	10000	30000	-7	1	0.6999
USAir	332	2126	-1658	1	0.6481



(a) Number of additional links in obtained graph H



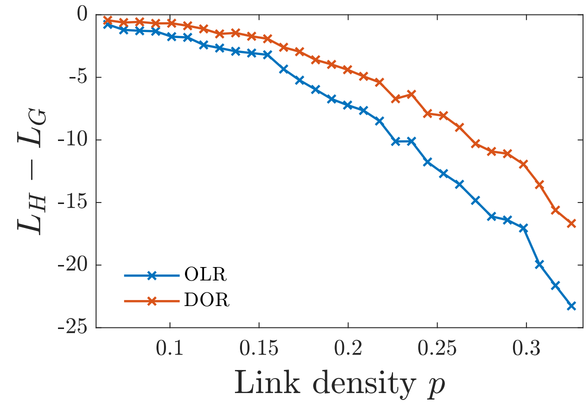
(b) Number of common links in G and H



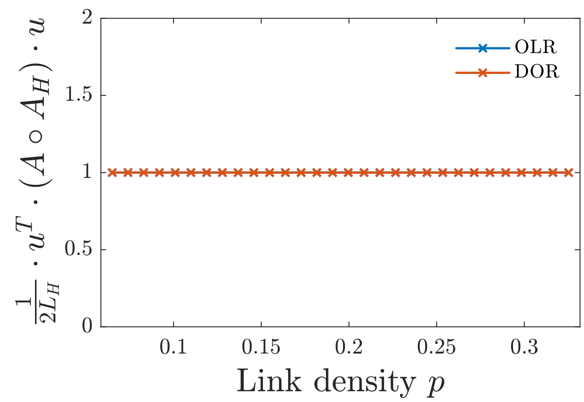
(c) Norm $||D-S||$

Fig. 11. Performance of the DOR and OLR on ER graphs with $N = 10$ nodes and different link density p. The input parameter $b = 0.7$.

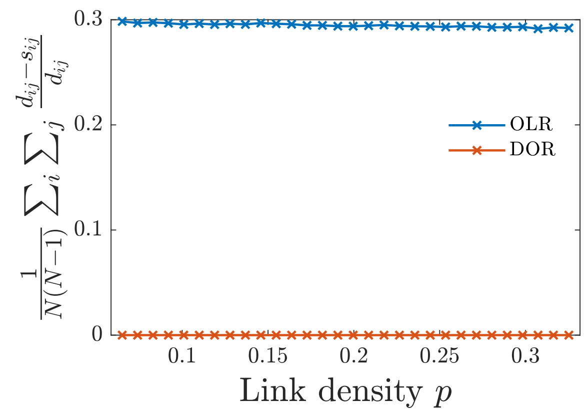
Table 2 and Table 3 respectively show the performances of DOR and OLR in BA network, WS network and an empirical network USAir. The node number N and link number L are shown in the tables. For each graph, the shortest path weight matrix are computed as the input demand matrix D for DOR and OLR. The input parameter of OLR $b = 0.3$. For



(a) Number of additional links in obtained graph H



(b) Number of common links in G and H



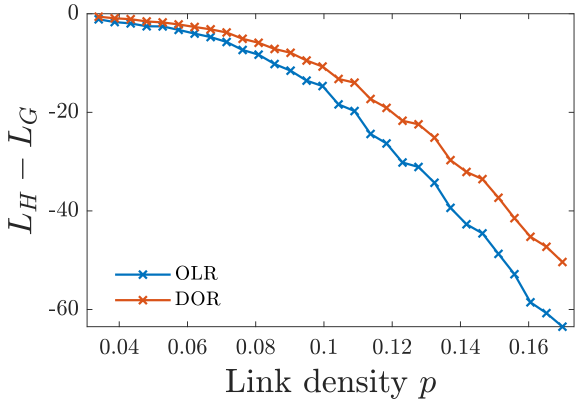
(c) Norm $||D-S||$

Fig. 12. Performance of the DOR and OLR on ER graphs with $N = 20$ nodes and different link density p. The input parameter $b = 0.7$.

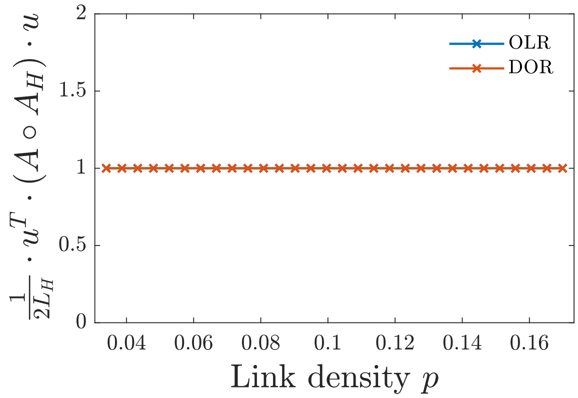
BA networks, the degree of node $\Pr[\mathrm{d} = \mathrm{k}] \propto \mathrm{k}^{-\gamma}$, where $\gamma = 2.75, 2.36, 2.48$ and 2.37 in BA1, BA2, BA3 and BA4, respectively. The link weights are uniformly distributed in $(0, 1)$. We generate WS networks as follows:

1) Create a ring lattice with N nodes of the mean degree $2k$.
2) Each node is connected to its k nearest neighbors on either side.
3) For each edge in the graph, rewire the target node with probability $p = 0.5$

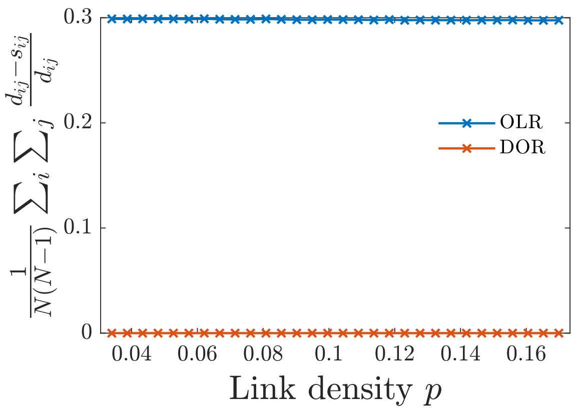
For WS1, WS2, WS3 and WS4, the corresponding $k = 2, 3, 3$ and 3, respectively. The link weights are uniformly distributed in $(0, 1)$.



(a) Number of additional links in obtained graph H



(b) Number of common links in G and H



(c) Norm $||D - S||$

Fig. 13. Performance of the DOR and OLR on ER graphs with $N = 50$ nodes and different link density p. The input parameter $b = 0.7$.

References

[1] P. Van Mieghem, "A tree realization of a distance matrix: the inverse shortest path problem with a demand matrix generated by a tree," *Delft University of Technology, Report20211012*, pp. 1–15, 2021.

[2] K. Devriendt and P. Van Mieghem, "The simplex geometry of graphs," *Journal of Complex Networks*, vol. 7, no. 4, pp. 469–490, 2019.

[3] M. Fiedler, *Matrices and graphs in geometry*. Cambridge, U.K.: Cambridge University Press, 2011.

[4] L. Mirsky, *An introduction to linear algebra*. Dover Publications, Inc., 1982.

[5] S. P. Fekete, W. Hochstättler, S. Kromberg, and C. Moll, "The complexity of an inverse shortest path problem," in *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future*, vol. 49, 1999, pp. 113–127.

[6] B. A. Miller, Z. Shafi, W. Ruml, Y. Vorobeychik, T. Eliassi-Rad, and S. Alfeld, "Pathattack: Attacking shortest paths in complex networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2021, pp. 532–547.

[7] D. Burton, W. Pulleyblank, and P. L. Toint, "The inverse shortest paths problem with upper bounds on shortest paths costs," in *Network Optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 156–171.

[8] D. Burton and P. L. Toint, "On an instance of the inverse shortest paths problem," *Mathematical programming*, vol. 53, no. 1, pp. 45–61, 1992.

[9] M. R. Garey and D. S. Johnson, *Computers and intractability*. New York: W. H. FREEMAN AND COMPANY, 1979.

[10] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 74, pp. 47–97, Jan 2002.

[11] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[12] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015.