Weighted chained graphs and some applications

C. Fenu[a], L. Reichel[b], G. Rodriguez[a], Y. Zhang[c]

[a] *Department of Mathematics and Computer Science, University of Cagliari, via Ospedale 72, Cagliari, 09124, Italy*
[b] *Department of Mathematical Sciences, Kent State University, Kent, 44242, OH, USA*
[c] *School of Data Science and Artificial Intelligence, Dongbei University of Finance and Economics, Dalian, 116025, Liaoning, China*

Abstract

This paper introduces weighted chained graphs, as well as minimal broadcasting and receiving sets, and investigates their properties. Both directed and undirected graphs are considered. The notion of central nodes is introduced both for weighted directed and undirected graphs. This notion is helpful for determining how quickly information can propagate throughout a graph. In particular, it is useful for the investigation of transportation networks and for city planning. Applications to the analysis of airline and bus networks are presented.

Keywords: network analysis, weighted chained graph, broadcasting set, receiving set, central vertex

---

1. Introduction

Many complex systems can be modeled as networks. A network is a set of objects, referred to as *nodes* or *vertices*, that are connected by *edges*. The nature of the nodes and edges depends on the application. Networks are represented by graphs, which typically leave out many details of the system they model. Nevertheless, graphs often are able to capture much of the complexity of the original system, and their relative simplicity makes them amenable to analysis. For instance, network models typically allow us to assess which nodes and edges are particularly important in a network. Graph models are employed in a wide range of areas including telecommunication, transportation, epidemiology, and biology; see, e.g., Bapat [1], Bogatti [2], De la Cruz Cabrera et al. [5], Estrada [7], Fenu and Higham [8], Newman [9], as well as Noschese and Reichel [10] for discussions of graphs and many applications.

We consider networks that can be represented by a weighted graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$, where $\mathcal{V} = \{v_i\}_{i=1}^n$ denotes a set of vertices or nodes, $\mathcal{E} = \{e_i\}_{i=1}^m$ is the set of edges that connects the vertices, and $\mathcal{W} = \{w_i\}_{i=1}^m$ are edge weights, i.e., the edge e_i is equipped with the weight w_i. Each weight w_i is a non-negative scalar that indicates the strength (or importance) of the connection between the vertices that are connected by the edge e_i. A graph is said to be *unweighted* if all positive weights w_i equal 1. We will consider simple graphs, i.e., graphs without multiple edges and self-loops. A *directed edge* e_k that points from vertex v_i to v_j can be identified with the ordered pair $e_k = (v_i, v_j)$. A directed edge may be considered a "one-way street". The number of nodes that can be reached from a specific node by a directed edge is the *out-degree* of that node, and the number of nodes that can reach a specific node by a directed edge is the *in-degree* of that node. An *undirected edge* e_k between the vertices v_i and v_j is identified with the set $e_k = \{v_i, v_j\}$. Thus, an undirected edge may be regarded as a "two-way street". If all the edges of a graph are undirected, then the graph is said to be *undirected*; otherwise the graph is directed. The number of edges that are incident to a specific node in an undirected graph is the *degree* of that node.

A *walk* with $k+1$ vertices is a sequence of vertices $v_{i_1}, v_{i_2}, \ldots, v_{i_{k+1}}$ and an associated sequence of edges $e_{i_1}, e_{i_2}, \ldots, e_{i_k}$ such that the edge e_{i_j} in this walk points from vertex v_{i_j} to vertex $v_{i_{j+1}}$, for $j = 1, 2, \ldots, k$. An undirected edge e_{i_j} in this walk is said to be between the vertices v_{i_j} and $v_{i_{j+1}}$. The length of the walk determined by the vertices $v_{i_1}, v_{i_2}, \ldots, v_{i_{k+1}}$ and edges $e_{i_1}, e_{i_2}, \ldots, e_{i_k}$ is defined as the sum of the weights of the edges in the walk, i.e., the length is given by $\sum_{j=1}^k w_{i_j}$. In particular, for an unweighted graph, the length of a walk is the number of edges that make up the walk. Vertices and edges of a walk may be repeated. A *path* is a walk in which no vertex is repeated. Assume that there is a path from node v_i to node v_j. Then the *distance* $d(v_i, v_j)$ from node v_i to node v_j is the length of the shortest path from node v_i to node v_j measured by the sum of the weights of the edges of the path. If the graph is unweighted, then $d(v_i, v_j)$ is the number of edges in the shortest path from v_i to v_j. Note that $d(v_i, v_j)$ may differ from $d(v_j, v_i)$; in fact, some distances might not be defined.

Recently, Concas et al. [3, 4] introduced the notions of chained directed and chained undirected graphs and some generalizations. For an undirected graph, the notion of a chained graph generalizes bipartivity and allows the determination of *central nodes* of the graph; see [4]. The analysis is based on the use of spanning trees for the graph. A generalization to directed graphs is described in [3]. Under suitable conditions, the chained structure can be uncovered by using spanning trees for directed graphs. When applicable, this analysis allows the definition of central nodes, and has been used to shed light on the structure of graphs that arise in a variety of applications; see [3]. However, some directed graphs do not have a directed spanning tree, and then the approach to define central nodes of an undirected graph proposed in [4] cannot be applied.

It is the purpose of this paper to generalize the approach to chained structure for directed graphs presented in [3] in several ways: We base our analysis on spanning forests instead of on spanning trees. This allows us to identify a chained structure, if present, for a general directed graph. Moreover, we allow edge weights different from one. This generalizes results both in [3, 4] and allows us to define weighted chained structures both for directed and undirected graphs.

This paper is organized as follows. Section 2 generalizes results on chained structure for unweighted undirected graphs described in [4] to weighted directed graphs. Our approach to define chained structure is based on the application of directed forests. This allows the definition of chained structure for a larger set of directed graphs than the approach in [3]. Moreover, the graphs are allowed to have edge weights different from unity. Section 3 defines the notions of node centrality for directed and weighted graphs. Broadcasting and receiving sets are introduced. This allows the definitions of out-centrality and in-centrality of nodes. Section 4 is concerned with the special case of weighted undirected graphs and generalizes the discussion in [4] by allowing weights different from unity. Some applications are presented in Section 5, and concluding remarks can be found in Section 6.

2. Chained structure of weighted directed graphs

The following definition extends the notion of chained graphs introduced in [3].

Definition 2.1. *A weighted directed graph (in short, "digraph") $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is said to be weighted* ℓ-chained *with initial vertex set $\mathcal{V}_1$ if to each edge $e \in \mathcal{E}$ is associated a positive weight $w \in \mathcal{W}$ and the set of vertices can be partitioned* *into ℓ disjoint non-empty subsets*

$$\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2 \cup \cdots \cup \mathcal{V}_\ell \tag{2.1}$$

such that all edges connect vertices in the set $\mathcal{V}_j$ to vertices in the set $\mathcal{V}_{j+1}$, for $j = 2, 3, \ldots, \ell - 1$. The chain length ℓ is the largest number of vertex subsets $\mathcal{V}_j$ possible with this property. The vertex sets $\mathcal{V}_j$ and $\mathcal{V}_{j+1}$, for $j = 1, \ldots, \ell - 1$, are said to be consecutive.

An undirected edge in a digraph may be regarded as a pair of directed edges of opposite orientations. Many real-world weighted networks may not have an ℓ-chained structure. We therefore also consider more general networks, described by Definition 2.2 below, that allow edges between non-consecutive vertex subsets.

Definition 2.2. *A weighted digraph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is said to be weighted (ℓ, k)-chained with initial vertex set $\mathcal{V}_1$ if it has the chained structure described in Definition 2.1 with the extension that edges may connect vertices belonging to non-consecutive vertex subsets. The* lower bandwidth *k is defined as the largest integer for which there exists an edge from a vertex in the subset $\mathcal{V}_j$ and a vertex in the subset $\mathcal{V}_{j-k}$.*

Remark 2.1. *The bandwidth k is always larger than or equal to -1 and smaller than ℓ. If $k = -1$, then Definition 2.2 agrees with Definition 2.1. If an edge connects a node v_1 in $\mathcal{V}_j$ to a node v_2 in $\mathcal{V}_{j+k}$, with $k > 1$, then node v_2 is to be moved to the set $\mathcal{V}_{j+1}$.*

A digraph is said to be *strongly connected* if there exist directed paths connecting each vertex pair (v_i, v_j) in both directions. It is *semi-connected* if for some vertex pair such a connection exists only in one direction. A digraph is *weakly connected* if there is an undirected path that connects any vertex pair (v_i, v_j), i.e., a path obtained by replacing all directed edges by undirected ones. An undirected graph is said to be *connected* if each vertex pair is connected by a path; see [7, 9].

Theorem 2.1. *Every weakly connected digraph is (ℓ, k)-chained.*

Proof. Let the node set $\mathcal{V}_1$ initially only contain the node v_1, and consider undirected paths for all vertex pairs (v_1, v_i), $i = 2, \ldots, n$. Let one of these paths, of length $s-1$, be $(v_1, v_{i_2}, \ldots, v_{i_{s-1}}, v_{i_s})$. If there is a directed edge that connects v_1 to v_{i_2}, then $v_{i_2} \in \mathcal{V}_2$; otherwise the edge is directed from v_{i_2} to v_1 and $v_{i_2} \in \mathcal{V}_0$. Proceed similarly for all other nodes in the path. If node v_{i_r} in the path already has been assigned to a set $\mathcal{V}_j$, then the node $v_{i_{r+1}}$ must be attributed either to the set $\mathcal{V}_{j-1}$ or to the set $\mathcal{V}_{j+1}$, depending on the orientation of the edge that connects the two vertices. The proof is valid for both weighted and unweighted directed graphs.

Since the graph is weakly connected, every node will be assigned to a set $\mathcal{V}_j$ and, given the way the sets have been constructed, every node in $\mathcal{V}_j$ will be connected to a node in $\mathcal{V}_{j+1}$. When the process ends, the node sets must be renumbered as $\mathcal{V}_1, \ldots, \mathcal{V}_\ell$, and the missing edges must be added to the new graph. If the initial graph is ℓ-chained, then we will end up with the chained structure of the graph. Otherwise, there will be some connections going from nodes in $\mathcal{V}_j$ to $\mathcal{V}_{j-k}$. If some k is smaller than -1, then the corresponding node must be relocated so that $k \geq -1$. In the end, the largest value of k will identify the underlying (ℓ, k)-chained structure. □

The following example explains how to detect an (ℓ, k)-chained structure of a weakly connected graph.

Example 2.1. *Consider the weakly connected digraph shown in Figure 1(a). To detect its (ℓ, k)-chained structure, let $v_1 \in \mathcal{V}_1$ be the initial node. Since there is a directed edge e_1 that points from v_1 to v_2, node v_2 should be placed in vertex set $\mathcal{V}_2$. Edge e_2 provides a connection from v_3 to v_2. Therefore, node v_3 belongs to vertex set $\mathcal{V}_1$. Continuing the process, nodes v_4 and v_5 are assigned to $\mathcal{V}_2$. Then, for the last node v_6, we have two possible assignments. We may assign v_6 to the vertex set $\mathcal{V}_1$ according to the direction of edge e_5. The edge e_6, which points from v_4 to v_6, indicates that $k = 1$ and a $(2,1)$-chained graph is achieved; see Figure 1(b).*

Alternatively, we may consider the direction of edge e_6 and assign v_6 to $\mathcal{V}_3$, as in Figure 1(c). However, this configuration is not permitted, as there is a "long" forward connection from v_3 to v_6. Hence, v_6 has to be moved to the set $\mathcal{V}_2$. Then, edges e_5 and e_6 indicate that $k = 0$, and we obtain the $(2,0)$-chained graph of Figure 1(d). Considering the edge e_7 would result in the same node sets.

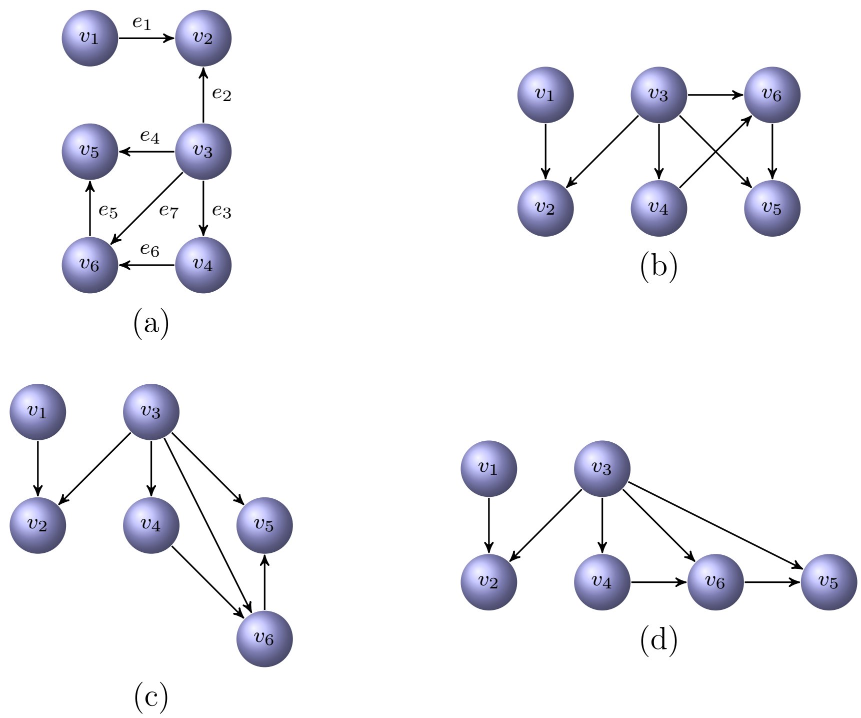

Figure 1: A weakly connected digraph (a) and two (ℓ, k)-chained structures (b and d) obtained from it.

Theorem 2.2. *Every digraph is (ℓ, k)-chained.*

Proof. Since any digraph is the union of weakly connected components, Theorem 2.1 can be applied to every one of these components. The chained structures of these components can be merged. For instance, let two components produce the chained structures $(\mathcal{V}_1, \ldots, \mathcal{V}_\ell)$ and $(\mathcal{V}'_1, \ldots, \mathcal{V}'_{\ell'})$, with $\ell' < \ell$. Then the first structure can be updated, without increasing the chain length, by setting $\mathcal{V}_i = \mathcal{V}_i \cup \mathcal{V}'_{i-j}$, $i = j+1, \ldots, j+\ell'$, for any $j = 0, \ldots, \ell - \ell'$. □

The chained structure of a directed graph can be identified independently of the weights. For the sake of simplicity, we drop the weights in the following when they are not relevant.

Definition 2.3. *A digraph is said to be an* out-tree *if it is acyclic (i.e., it does not contain any directed or undirected cycles) and if it has only one vertex v with zero in-degree. The vertex v is called the* root *of the out-tree.*

When the graph is undirected, the in-degree and out-degree of a node are the same, and are referred to as the degree of the node.

Sometimes, an out-tree is referred to as an *arborescence*, or a *branching*; see [6]. Figure 2 shows an example of an arborescence.

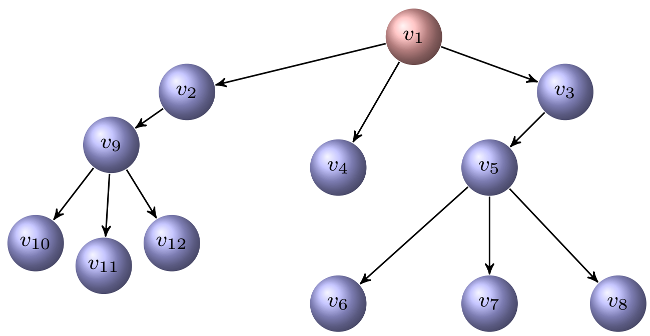

Figure 2: An arborescence rooted at node v_1.

Definition 2.4. *A digraph is said to be an* in-tree *if it is acyclic and only has one vertex v with zero out-degree. This vertex is referred to as the* root *of the in-tree.*

Definition 2.5. *A* spanning subgraph $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}'\}$ *of a weakly connected digraph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is a graph with the same vertex set $\mathcal{V}$ but a possibly smaller edge set $\mathcal{E}' \subset \mathcal{E}$. A spanning out-tree (in-tree) is a spanning subgraph that is an out-tree (in-tree).*

Figure 3 shows an example of a digraph that admits a spanning out-tree rooted at node v_1.

Not all digraphs admit a spanning out-tree or in-tree. These digraphs can be studied with the aid of spanning forests.

Definition 2.6. *A* spanning forest $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}'\}$ *for a digraph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is an acyclic spanning subgraph that may consist of disconnected components. Every connected component of a spanning forest is either an out-tree or an in-tree.*

Every spanning forest is a weighted ℓ-chained graph with the vertex set $\mathcal{V}_1$ containing the roots of each arborescence and the subset $\mathcal{V}_i$ containing the

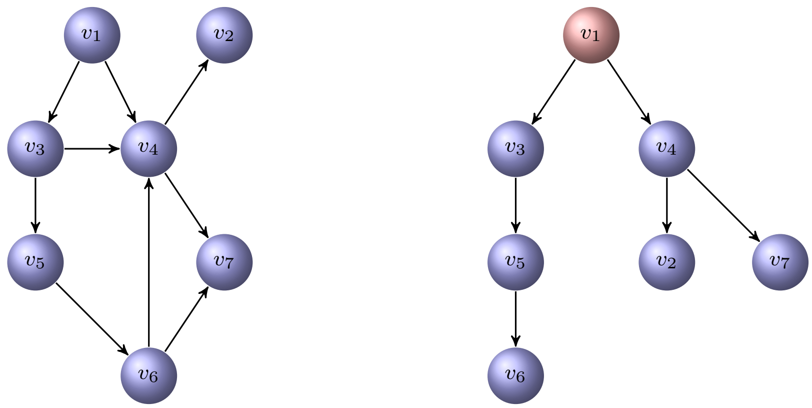

Figure 3: A digraph (left) and one of its spanning arborescences (right).

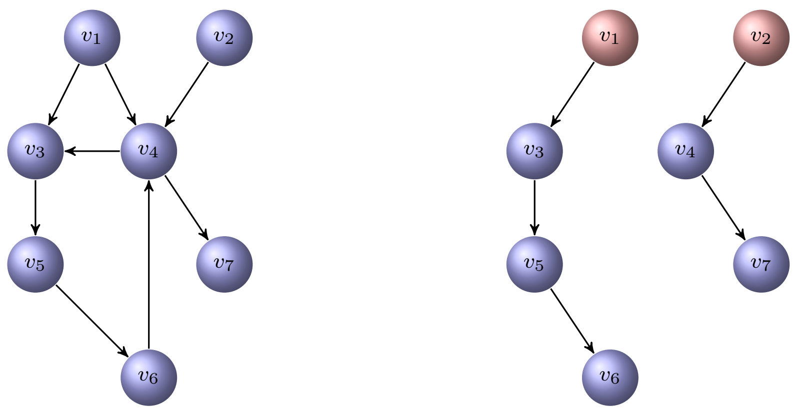

Figure 4: On the left, a digraph that does not admit a spanning arborescence; on the right, a spanning forest for the graph.

children of the vertices in $\mathcal{V}_{i-1}$, for $i = 2, \ldots, \ell$. The roots of each out-tree may be placed in node sets different from $\mathcal{V}_1$, if this is useful.

A chained digraph is not necessarily a forest, but any (ℓ, k)-chained digraph has a spanning forest. Figure 4 shows an (ℓ, k)-chained graph which is not a forest and does not admit a spanning out-tree, together with a spanning forest for it. The roots of each out-tree in the forest are marked in red. This digraph is $(4, 2)$-chained with $\mathcal{V}_1 = \{v_1, v_2\}$, $\mathcal{V}_2 = \{v_3, v_4\}$, $\mathcal{V}_3 = \{v_5, v_7\}$, and $\mathcal{V}_4 = \{v_6\}$; the bandwidth is 2 since there is a connection from $v_6 \in \mathcal{V}_4$ to $v_4 \in \mathcal{V}_2$. Figure 5 displays other spanning forests for the same graph with roots v_1 and v_2.

The (ℓ, k)-chained structure of a weakly connected graph is closely related

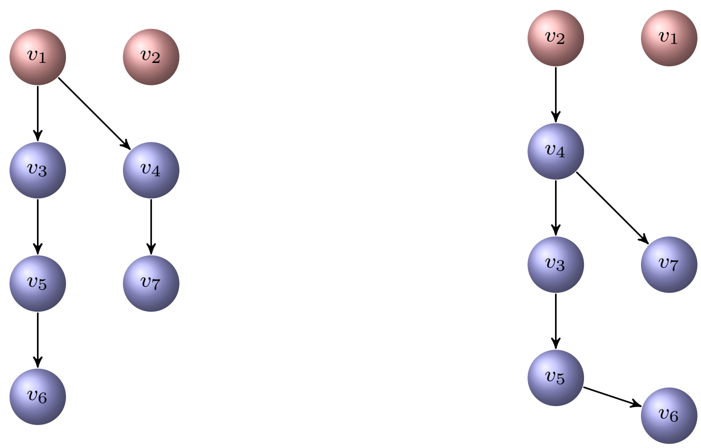

Figure 5: Spanning forests of the digraph in Figure 4.

to a spanning forest for the graph, in the sense that the latter can be deduced from the former. On the contrary, a spanning forest alone does not allow the determination of the chained structure for the graph that it spans, because the spanning forest lacks information about which sets $\mathcal{V}_j$ contain the roots.

Motivated by the above discussion, we show in the following that spanning trees and forests are useful tools for detecting the chained structure of a digraph. Algorithm 1 constructs a forest containing a subset of the nodes either made of out-trees or of in-trees, depending on the value of the third input argument. The result is simply a tree if a single node is given as input. If an incomplete chained structure for a network is given as input, then Algorithm 1 extends this structure by constructing either all the out-trees by starting from nodes in $\mathcal{V}_1$ or all in-trees ending at nodes in $\mathcal{V}_\ell$. The algorithm starts by scanning the input chained structure by considering all the available node sets $\mathcal{V}_j$, $j = 1, \ldots, \ell$, or initializing a new set as empty; see lines 5–9. If an out-forest is sought, then the algorithm selects all the nodes pointed to by vertices in $\mathcal{V}_{j-1}$, removing those already present in another set (lines 10–18), and adds them to the set $\mathcal{V}_j$. A slight modification of Algorithm 1 allows the handling of an in-forest. The iteration is interrupted when all the nodes have been added (in this case a spanning forest is obtained) or when no new nodes are available. The output is the updated chained structure and the corresponding forest.

Algorithm 2 identifies the chained structure of a graph, returning the node sets $\mathcal{V}_i$, $i = 1, \ldots, \ell$, and the corresponding spanning forest. It starts

Algorithm 1 Construction of an out-forest or a in-forest (function IOFOREST).

Require: Adjacency matrix $A = [a_{ij}]_{i,j=1}^{n} \in \mathbb{R}^{n \times n}$, incomplete directed chained structure $\mathcal{V}$, inflag = 'in' or 'out'
Ensure: Out/in forest $\mathcal{F} = \{\mathcal{T}_1, \ldots, \mathcal{T}_\tau\}$ starting/ending at the first/last set of $\mathcal{V}$, updated chained structure $\mathcal{V}$, set $\mathcal{S}$ of the nodes involved

if inflag = 'in' **then** reverse the chained structure $\mathcal{V}$
$\mathcal{S} = \cup \mathcal{V}_i$, $\ell = \text{length}(\mathcal{V})$, $j = 1$
while cardinality($\mathcal{S}$) $< n$ **do**
    $j = j + 1$
    if $j \leq \ell$ **then**
        $V = \mathcal{V}_j$
    else
        $V = \emptyset$
    end if
    for $v \in \mathcal{V}_{j-1}$ **do**
        if inflag = 'in' **then**
            $\Omega = \{$nodes pointing to $v\}$
        else
            $\Omega = \{$nodes pointed by $v\}$
        end if
        $\Omega = \Omega \setminus (\Omega \cap \mathcal{S})$ (remove from Ω the nodes contained in $\mathcal{S}$)
        $V = V \cup \Omega$, $\mathcal{S} = \mathcal{S} \cup \Omega$
    end for
    if $V = \emptyset$ **and** $j > \ell$ **then**
        $j = j - 1$
        exit the **while** loop
    end if
    $\mathcal{V}_j = V$
end while
Extract an out/in forest $\mathcal{T}$ from $\mathcal{V}$
if inflag = 'in' **then** reverse the chained structure $\mathcal{V}$

by constructing, in line 1, the out-tree starting at a chosen node v. This produces an initial partial chained structure $\mathcal{V}$. Then the algorithm extends this structure (see lines 4–9) by iteratively constructing the in-forest ending at $\mathcal{V}_\ell$ and the out-forest starting at $\mathcal{V}_1$. This is done by calling Algorithm 1.

Algorithm 2 terminates when all the nodes have been included in the chained structure. If this fails, then the graph is not weakly connected; this follows from Theorem 2.1. If needed, then the chained structure obtained can be extended by initializing Algorithm 2 with a node that is not in the connected component just found. The chained structures so determined may be joined by identifying the node sets with the same index, but different couplings are possible; see the proof of Theorem 2.2. We remark that Algorithms 1 and 2 do not perform any floating-point operations, only integer and set operations. Their time complexity can be characterized by the fact that the n vertices and m edges are visited at least once.

Algorithm 2 Identification of a directed (ℓ, k)-chained graph.

Require: Adjacency matrix $A = [a_{ij}]_{i,j=1}^n \in \mathbb{R}^{n\times n}$, initial node v

Ensure: Directed chained structure $\mathcal{V} = \{\mathcal{V}_1, \dots, \mathcal{V}_\ell\}$ and spanning forest $\mathcal{F} = \{\mathcal{T}_1, \dots, \mathcal{T}_\tau\}$, if they exist

$[\mathcal{F}, \mathcal{V}, \mathcal{S}] = \text{IOFOREST}(A, v, \text{'out'})$ *% determine an out-forest $\mathcal{F}$, with root at node v, and the corresponding chained structure $\mathcal{V}$ involving the nodes in $\mathcal{S}$*

$N_{\text{old}} = -1$

$N = \text{length}(\mathcal{S})$

while $N < n$ and $N \neq N_{\text{old}}$ **do**

  $N_{\text{old}} = N$

  $[\mathcal{F}, \mathcal{V}, \mathcal{S}] = \text{IOFOREST}(A, \mathcal{V}, \text{'in'})$ *% find an in-forest ending at $\mathcal{V}_\ell$*

  $[\mathcal{F}, \mathcal{V}, \mathcal{S}] = \text{IOFOREST}(A, \mathcal{V}, \text{'out'})$ *% find an out-forest starting at $\mathcal{V}_1$*

  $N = \text{length}(\mathcal{S})$

end while

if $N < n$ **then**

  The graph has not a chained structure, so it is not weakly connected.

end if

3. Broadcasting sets, receiving sets, and central nodes for directed graphs

It is important in some applications to be able to determine a small set of nodes of a digraph that can spread information to all other nodes of a network at minimal cost, where the cost is measured by summing the weights of the edges traversed. We will denote such a set as the *minimum broadcasting set.*

Chained structures and spanning forests are helpful for determining such sets.

Definition 3.1. *A* minimum spanning forest *is a spanning forest whose sum of weights is the minimum possible.*

Definition 3.2. *A* broadcasting set *for a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is a subset of nodes $\mathcal{B} \subset \mathcal{V}$ such that*

1. *$\mathcal{B}$ is connected to every other node in $\mathcal{V}$, in the sense that for any $v_j \notin \mathcal{B}$ there is a node $v_i \in \mathcal{B}$ connected by a path to v_j;*
2. *$\mathcal{B}$ is "essential", that is, by removing any node from $\mathcal{B}$ property 1 is lost.*

A minimum broadcasting set *is such that the distance between $\mathcal{B}$ and any other vertex in the network*

$$\epsilon^{\text{out}}(\mathcal{B}) = \max_{v_j \notin \mathcal{B}} \min_{v_i \in \mathcal{B}} d(v_i, v_j)$$

is minimal.

Similarly, a receiving set *for a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is an "essential" subset $\mathcal{R} \subset \mathcal{V}$ such that there is a path from every node in $\mathcal{V}$ ending in $\mathcal{R}$. A* minimum receiving set *is such that the distance between the vertices in the network and $\mathcal{R}$,*

$$\epsilon^{\text{in}}(\mathcal{R}) = \max_{v_j \notin \mathcal{R}} \min_{v_i \in \mathcal{R}} d(v_j, v_i),$$

is minimal.

The *eccentricity* of a vertex in a graph is the maximum distance from this vertex to any other vertex of the graph. Consider a broadcasting set as a macro-node. Then $\epsilon^{\text{out}}(\mathcal{B})$ coincides with the *eccentricity* of the broadcasting set. By minimizing the eccentricity, the radius of $\mathcal{G}$ is obtained. We will refer to $\epsilon^{\text{out}}(\mathcal{B})$ and $\epsilon^{\text{in}}(\mathcal{R})$ as the out-eccentricity of $\mathcal{B}$ and in-eccentricity of $\mathcal{R}$, respectively.

If a network is ℓ-chained, then there is only one broadcasting set. It coincides with $\mathcal{V}_1$ plus any other node without an incoming edge.

Theorem 3.1. *A broadcasting set for a general weakly connected graph is always the union of a subset of the set $\mathcal{V}_1$ of an (ℓ, k)-chained structure for the graph, and the set of nodes with zero in-degree.*

Proof. The subset of $\mathcal{V}_1$ is obtained by removing nodes with zero out-degree. $\square$

An analogous statement can be made for the receiving set $\mathcal{R}$ and the set $\mathcal{V}_\ell$. In this case, nodes with zero out-degree must be included in $\mathcal{R}$. In an (ℓ, k)-chained network, different chained structures can be determined. It is therefore meaningful to consider the minimum broadcasting set.

Example 3.1. *The graph in Figure 6(a) admits three (ℓ, k)-chained structures ((b),(c), and (d) in Figure 6). The broadcasting sets associated with these chained structures are different. From the $(4, 2)$-chained structure (b), the obtained broadcasting set is $\mathcal{B}_b = \{v_1, v_5\}$. Similarly, from the $(3, 2)$-chained structure (c) and the $(3, 2)$-chained structure (d), we obtain $\mathcal{B}_c = \{v_2, v_5\}$ and $\mathcal{B}_d = \{v_3, v_5\}$, respectively.*

To better illustrate the connections between a broadcasting or receiving set and the rest of the vertices in a digraph, we introduce the concept of centrality to describe the spread of information from the broadcasting set or the reception of information by the receiving set.

Definition 3.3. *The* out-centrality *of the broadcasting set $\mathcal{B}_i$ is defined as*

$$P_p^{\text{out}}(\mathcal{B}_i) = \left(\sum_{v_j \notin \mathcal{B}_i} \min_{v_i \in \mathcal{B}_i} d(v_i, v_j)^p \right)^{1/p},$$

where $d(v_i, v_j)$ denotes the distance from node $v_i \in \mathcal{B}_i$ to node $v_j \in \mathcal{V} \setminus \mathcal{B}_i$, as defined in Section 1, and $p \in \mathbb{R}$. We refer to a broadcasting set $\mathcal{B}_m$ with the smallest out-centrality as a p-minimum broadcasting set*.*

Similarly, the in-centrality *of the receiving set $\mathcal{R}_i \in \mathcal{V}$ is given by*

$$P_p^{\text{in}}(\mathcal{R}_i) = \left(\sum_{v_j \notin \mathcal{R}_i} \min_{v_i \in \mathcal{R}_i} d(v_j, v_i)^p \right)^{1/p},$$

where $p \in \mathbb{R}$. The receiving set with the smallest in-centrality, $\mathcal{R}_m$, is called the p-minimum receiving set*.*

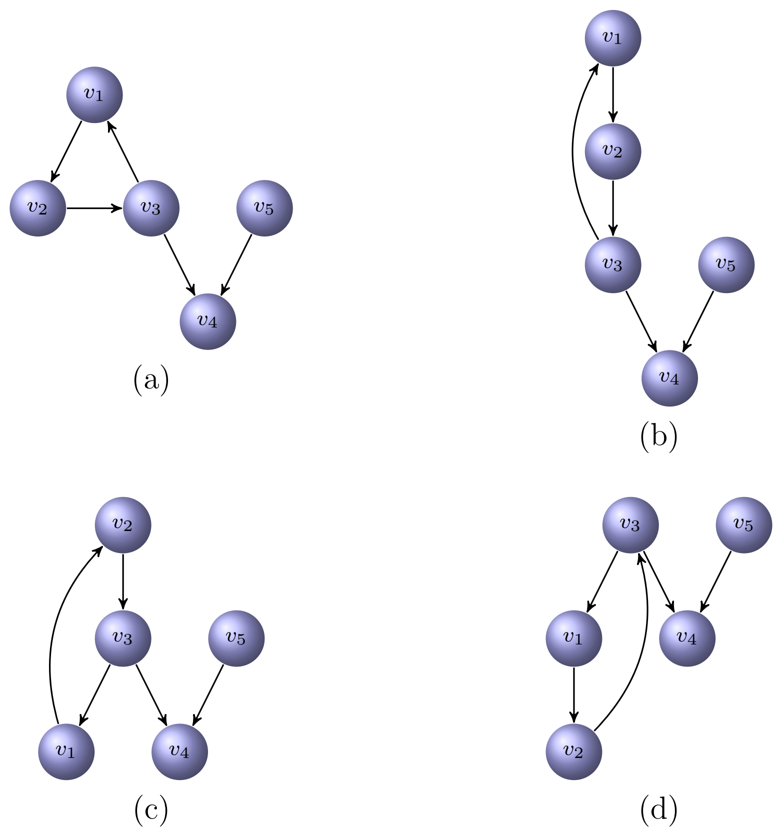

Figure 6: An unweighted weakly connected digraph with some of its (ℓ, k)-chained structures(b,c,and d)

Example 3.2. *Consider the graph in Figure 6(a). We identified in Example 3.1 the broadcasting sets $\mathcal{B}_b = \{v_1, v_5\}$, $\mathcal{B}_c = \{v_2, v_5\}$ and $\mathcal{B}_d = \{v_3, v_5\}$. After assigning weights to each edge in the graph as shown on the left of Figure 7, we let $p = 2$ and evaluate the out-centrality of the broadcasting set $\mathcal{B}_b$ as*

$$P_2^{\text{out}}(\mathcal{B}_b) = (2^2 + 5^2 + 5^2)^{1/2} \approx 7.348.$$

The out-centrality of $\mathcal{B}_c$ and $\mathcal{B}_d$ is calculated as $P_2^{\text{out}}(\mathcal{B}_c) = (6^2+3^2+5^2)^{1/2} \approx 8.367$ and $P_2^{\text{out}}(\mathcal{B}_d) = (3^2 + 5^2 + 2^2)^{1/2} \approx 6.164$, respectively. Therefore, $\mathcal{B}_d$ is the 2-minimum broadcasting set. Similarly, from Figure 6(b), we identify the receiving set $\mathcal{R}_b = \{v_4\}$, which is displayed on the right of Figure 7. The in-centrality of $\mathcal{R}_b$ with $p = 2$ is

$$P_2^{\text{in}}(\mathcal{R}_b) = (7^2 + 5^2 + 2^2 + 5^2)^{1/2} \approx 10.149.$$

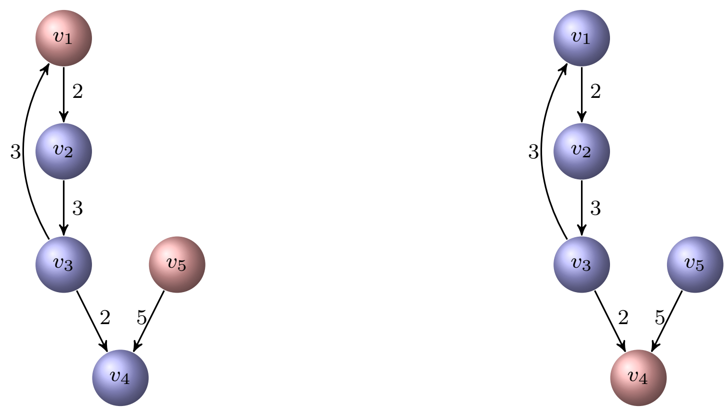

Figure 7: A weighted weakly connected digraph with the broadcasting set $\mathcal{B}_b = \{v_1, v_5\}$ on the left and the receiving set $\mathcal{R}_b = \{v_4\}$ on the right.

If we instead let $p = 1$, then we obtain $P_1^{\text{out}}(\mathcal{B}_b) = 12$, $P_1^{\text{out}}(\mathcal{B}_c) = 14$, $P_1^{\text{out}}(\mathcal{B}_d) = 10$, and $P_1^{\text{in}}(\mathcal{R}_b) = 19$.

Algorithm 3 summarizes the steps to obtain the broadcasting set $\mathcal{B}_i$ and its out-position centrality starting from the chained structure $\mathcal{V}$ with vertex v_i as a root. Once that all the broadcasting sets have been found, the one with smallest out-centrality easily can be selected.

4. Central nodes for undirected graphs

This section discusses how to determine the chained structure of weighted undirected graphs, and how to identify their center node(s). We consider graphs that are undirected, connected, simple, and weighted. The chained structure of these graphs extends the notion of chained graphs in [4].

The vertices v_i and v_j are said to be *adjacent* if there is an edge that connects these nodes.

Definition 4.1. *An undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is said to be weighted and* ℓ-chained *with initial vertex v_i if each edge $e \in \mathcal{E}$ is associated with a positive weight $w \in \mathcal{W}$ and the set of vertices $\mathcal{V}$ can be partitioned into ℓ disjoint non-empty subsets*

$$\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2 \cup \cdots \cup \mathcal{V}_\ell, \tag{4.1}$$

such that $v_i \in \mathcal{V}_1$ and all the vertices in the set $\mathcal{V}_j$ are adjacent only to vertices in the sets $\mathcal{V}_{j-1}$ or $\mathcal{V}_{j+1}$, for $j = 2, 3, \ldots, \ell - 1$. The chain length ℓ

Algorithm 3 Identification of the broadcasting set and computation of its out-centrality.

Require: Adjacency matrix $A = [a_{ij}]_{i,j=1}^n \in \mathbb{R}^{n\times n}$, (ℓ, k)-chained structure $\mathcal{V}$, $p \in \mathbb{R}$

Ensure: broadcasting set $\mathcal{B}_i$ and out-position centrality $P_p^{\text{out}}(\mathcal{B}_i)$

$\mathcal{B}_i = \mathcal{V}_1$

$\mathcal{T} = \mathcal{V} \setminus \mathcal{V}_1$

$D = \text{distances}(\mathcal{B}_i, \mathcal{T})$

$d = \min(D)$

if $\text{any}(d) = \infty$ **then** % *Some nodes in $\mathcal{T}$ cannot be reached from $\mathcal{B}_i$*

  $V = \text{isinf}(d)$ % *Find unreachable nodes*

  $\mathcal{A} = \mathcal{T}(V)$

  $\mathcal{A} = \mathcal{A} \setminus \{\text{nodes with zero out-degree}\}$

  $\mathcal{B}_i = \mathcal{B}_i \cup \mathcal{A}$

end if

$\mathcal{T} = \mathcal{T} \setminus \mathcal{A}$

$D = \text{distances}(\mathcal{B}_i, \mathcal{T})$

$d = \min(D)$

$P_p^{\text{out}} = \|d\|_p$

of the graph $\mathcal{G}$ is the largest number of vertex subsets $\mathcal{V}_j$ with this property. It typically depends on the choice of initial vertex v_i. The vertex sets $\mathcal{V}_j$ and $\mathcal{V}_{j+1}$ are said to be consecutive for $j = 1, 2, \ldots, \ell - 1$.

Definition 4.2. *An undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ is said to be weighted* ℓ-semi-chained *with initial vertex v_i if it has the chained structure described in Definition 4.1 with the extension that connections are allowed between vertices belonging to the same vertex subset.*

We remark that every weighted undirected graph has an ℓ-(semi-)chained structure. The determination of the chained structure of weighted undirected graphs is independent of the weights. The notion of a *tree* can be helpful for determining the chained structure of a graph.

Definition 4.3. *A* tree *is a connected undirected graph in which any two vertices are connected by exactly one path. Any vertex of a tree may be designated as the* root. *Vertices with degree one, except for the root, are referred to as* leaves.

Definition 4.4. *A* spanning tree *for a weighted undirected graph* $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ *is a subgraph* $\mathcal{T} = \{\mathcal{V}, \mathcal{E}', \mathcal{W}'\}$ *that is a tree and contains all the vertices of* $\mathcal{G}$.

Definition 4.5. *A* shortest-path spanning tree *is a spanning tree such that the path distance from the root to any other vertex is the smallest possible.*

We remark that the shortest-path spanning tree is not necessarily unique. The following example illustrates the process of determining the chained structure of a weighted undirected graph by identifying a shortest-path spanning tree.

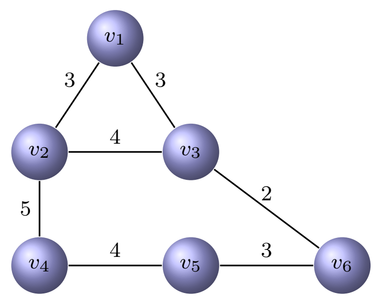

Figure 8: A weighted undirected graph.

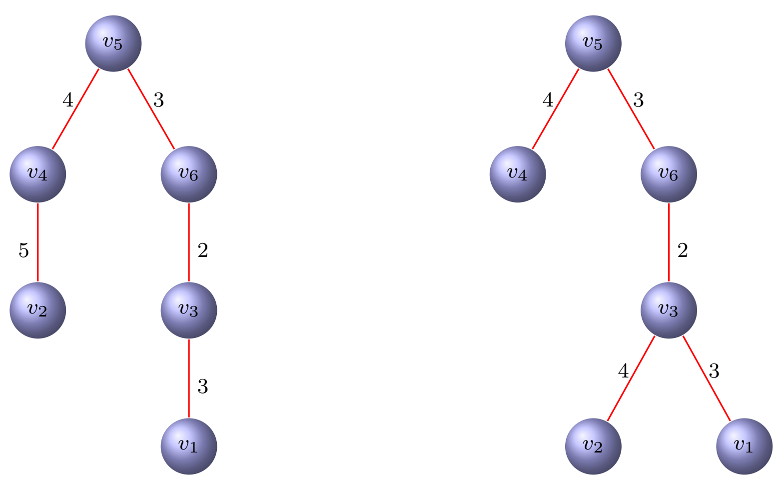

Figure 9: The shortest-path trees $\mathcal{T}_1^5$(left) and $\mathcal{T}_2^5$(right).

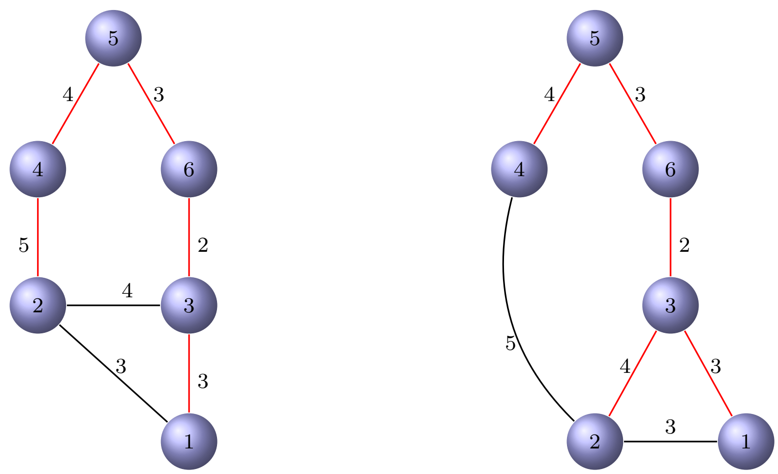

Figure 10: The missing edges (in black) are added to the trees in Figure 9 respectively.

Example 4.1. *Figure 9 shows two shortest-path spanning trees rooted at vertex v_5 for the weighted undirected graph $\mathcal{G}$ in Figure 8. Notice that from vertex v_5 to vertex v_2 there are two shortest paths of length 9. One path contains the edge from v_5 to v_4 and the edge from v_4 to v_2. This can be seen in the shortest-path spanning tree $\mathcal{T}_1^5$. The other path displays the shortest-path spanning tree $\mathcal{T}_2^5$, which connects nodes from v_5 to v_6, from v_6 to v_3, and from v_3 to v_2.*

The chained structures of the shortest-path trees $\mathcal{T}_1^5$ and $\mathcal{T}_2^5$ also can be identified from Figure 9. The partition of nodes of $\mathcal{T}_1^5$ is $\mathcal{V}_1 = \{v_5\}$, $\mathcal{V}_2 = \{v_4, v_6\}$, $\mathcal{V}_3 = \{v_2, v_3\}$, and $\mathcal{V}_4 = \{v_1\}$. For the tree $\mathcal{T}_2^5$, the partition is given by $\mathcal{V}_1 = \{v_5\}$, $\mathcal{V}_2 = \{v_4, v_6\}$, $\mathcal{V}_3 = \{v_3\}$, and $\mathcal{V}_4 = \{v_1, v_2\}$.

To determine the chained structure of the graph $\mathcal{G}$, we add the edges in $\mathcal{G}$, but not in $\mathcal{T}_1^5$, to the spanning tree $\mathcal{T}_1^5$ as shown on the left of Figure 10. The graph $\mathcal{G}$ is identified as a 4-semi-chained graph. Similarly, the graph on the right in Figure 10 is constructed by adding the missing edges in $\mathcal{G}$, but not in $\mathcal{T}_2^5$, to the spanning tree. With this node partitioning, the structure is not chained since the adjacent nodes v_4 and v_2 neither belong to two consecutive node subsets nor to the same node subset.

For weighted undirected graphs, it is interesting to determine nodes that can spread information to all the other nodes in the graph in the shortest amount of time, or for the least cost, depending on the meaning attributed to the weights of the edges. Such nodes are referred to as center nodes. The determination of center nodes can be easily achieved if the chained structure and the associated shortest-path spanning tree of a weighted undirected graph are known.

Definition 4.6. *Let $\mathcal{T}$ be a shortest-path spanning tree of the weighted undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$ with chained structure*

$$\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2 \cup \cdots \cup \mathcal{V}_\ell \tag{4.2}$$

starting at vertex $v_i \in \mathcal{V}_1$. The centrality of vertex v_i, for $p \geq 0$, is defined as

$$P_p(v_i) = \left(\sum_{j=1}^{n} d(v_i, v_j)^p \right)^{1/p}.$$

This centrality measure coincides with the p-norm of the vector of the distances between the vertex v_i and any other vertex in the graph when $p \geq 1$. We refer to a vertex with the smallest centrality as a p-center vertex.

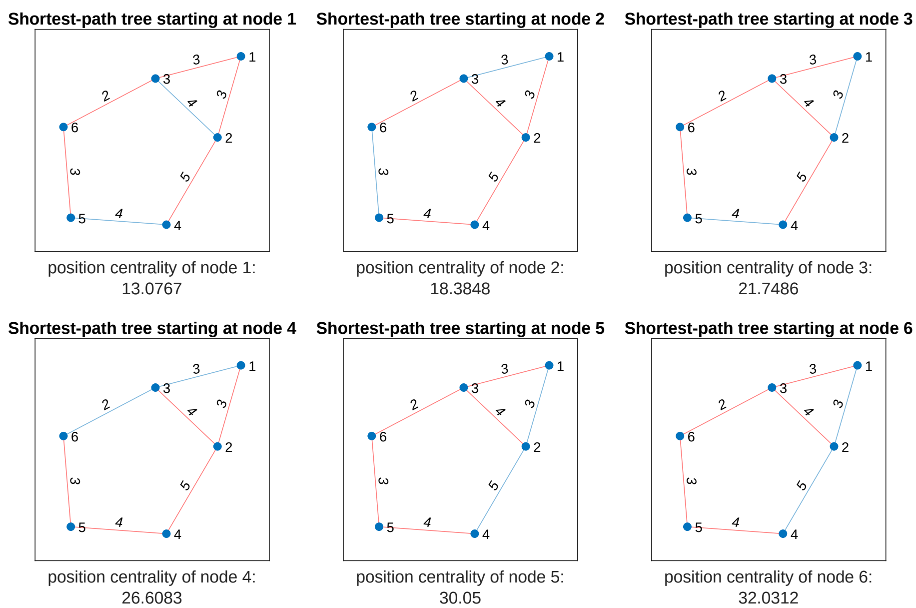

Figure 11: Shortest-path spanning trees rooted at each node of the weighted graph $\mathcal{G}$ in Figure 8, each with its computed position centrality for $p = 2$.

We remark that for unweighted undirected graphs, the definition of position centrality with $p = 1$ coincides with the one given in [4] for undirected chained graphs.

Example 4.2. *Consider the graph in Figure 8. The spanning trees rooted at each vertex v_i, for $i = 1, 2, \ldots, 6$, with its computed position centrality for $p = 2$ are displayed in Figure 11. The 2-center node is the node v_1.*

5. Some examples

This section illustrates how the broadcasting and receiving sets and its associated chained structure can be determined in real world transportation networks. We investigate the impact of weight changes on the out- and in-centralities of the broadcasting and receiving sets.

5.1. Airline network

We consider the airline data set reported by the Bureau of Transportation Statistics of the U.S. Department of Transportation. It describes the airline routes between 129 cities in the 48 contiguous states of the U.S. for the first three quarters of 2019. From this airline data set, we obtain a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$, where the node set $\mathcal{V}$ and the edge set $\mathcal{E}$ are represented by cities and airline routes, respectively. The entries of $\mathcal{W} = [w_{i,j}]_{i,j=1}^{129}$ denote weights that equal the number of passengers in the flights from city i to city j, for $i, j = 1, 2, \ldots, 129$. If there are no flights between two cities, the corresponding weight is set to zero. Since the number of passengers of the flights is reported for each quarter, we take the average numbers as the weights. Since the flights between some of the cities are one-way, the airline network is directed.

Let us first determine the broadcasting and receiving sets. We first identify the minimum broadcasting set, which contains 83 vertices with 1-out-centrality equal to 10973. The associated chained structure is $\{4, 2\}$-chained, which can be obtained when the initial vertex is one of the following:

$$v_{10},\ v_{28},\ v_{33},\ v_{49},\ v_{56},\ v_{60},\ v_{68},\ v_{78}.$$

The minimum receiving set contains 76 vertices and its 1-in-centrality is 12505. The associated chained structure is $\{4, 2\}$-chained with initial vertex v_{25}.

Figure 12: Some airports in the United States.

To enhance the visualization of the broadcasting and receiving sets, we use red and blue dots to represent the nodes in the minimum broadcasting set and the minimum receiving set, respectively, on the map of the United State in Figure 12. This map is available at `https://usamap360.com/usa-airports-map`. We notice that 30 vertices appear both in the broadcasting set and in the receiving set, which are displayed by green dots. Not all of the 129 cities are marked on the map.

It can be seen that the vertices marked by red and blue dots are mostly small and medium-sized cities, while most of the vertices represented by green dots are large cities. Among the vertices in the minimum receiving set, most of them are tourist destinations, such as San Diego and San Francisco in California, and Orlando, Tampa and West Palm Beach in Florida. We remark that nodes in the minimal broadcasting and receiving sets are not necessarily important airports; some of the airports in these sets are just difficult to reach or leave.

5.2. Bus network

This example considers the bus system that serves the metropolitan region surrounding the town of Cagliari in Sardinia, Italy. The region is roughly 100 km^2, with a population of $4.2 \cdot 10^5$ people, and is made up of the town of Cagliari as well as other smaller, sometimes contiguous municipalities that are very close to Cagliari, such as Monserrato, Selargius, Quartucciu, Quartu Sant'Elena, Elmas, Assemini, and Decimomannu; see Figure 13.

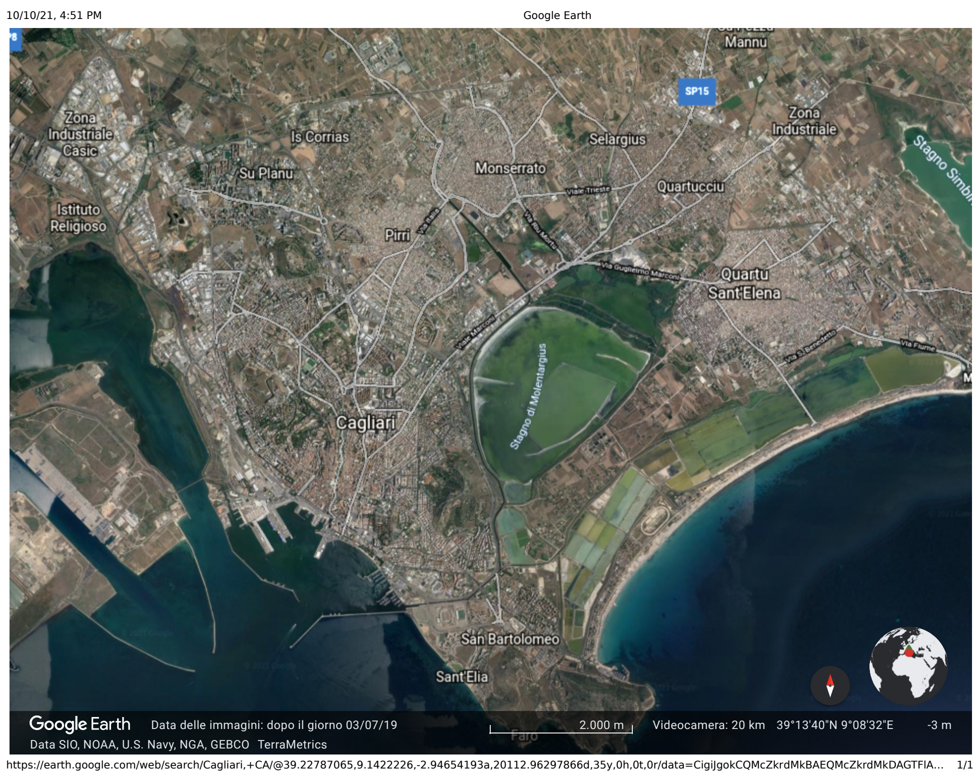

Figure 13: Cagliari metropolitan area; image produced by Google Earth.

This example has been studied in [3] as an unweighted undirected network, using only information about connections between bus stops. The data set used in this section contains information about the number of passengers transported, too. It therefore can be represented by a weighted graph.

Every bus stop defines a node in the network and bus routes between consecutive stops define edges. The network has 912 nodes and 1068 edges.

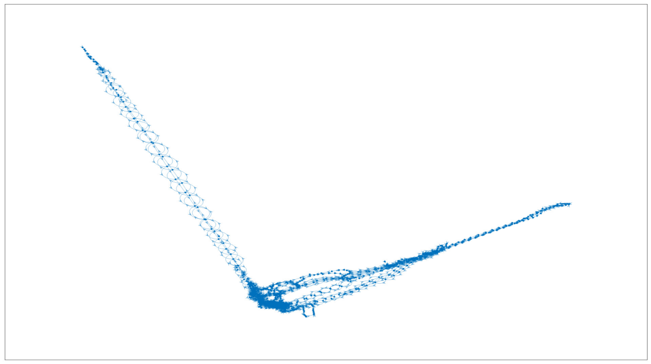

Figure 14: Bus network of the Cagliari metropolitan area.

Figure 14 shows a graphical representation of the network and Figure 15 shows the spy plot of the adjacency matrix A. The resulting network is weighted, that is, every edge is equipped with a nonnegative weight corresponding to the number of passengers traversing that edge in one day. Bus routes are direction-dependent since certain streets are one-way. Thus, the bus network is directed.

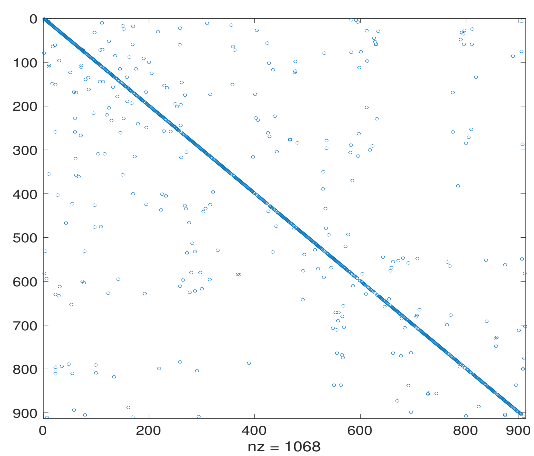

Figure 15: Spy plot of the adjacency matrix of the bus network of the Cagliari metropolitan area.

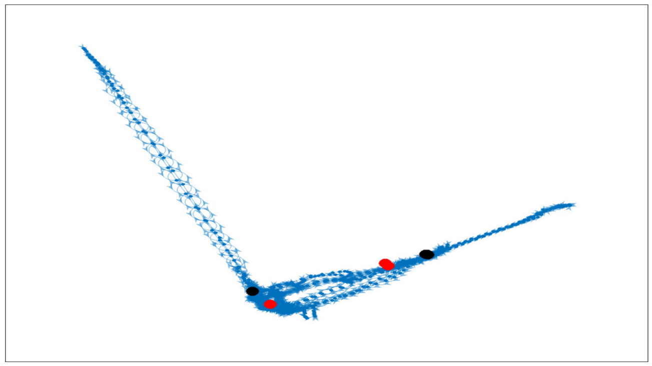

Figure 16: Bus network of the Cagliari metropolitan area. The red dots are the vertices in the minimum broadcasting set and the black dots are the ones in the minimum receiving set.

We first determine the broadcasting and receiving sets. The minimum broadcasting set contains 3 vertices, namely v_{223}, v_{704}, and v_{705}, with 1-out-centrality equal to $2.99 \cdot 10^7$. The associated chained structure is $\{70, 53\}$-chained which can be obtained when v_{223} is the initial vertex. The minimum receiving set contains the vertices v_{355}, v_{700}, v_{701}, v_{702}, v_{703}, and v_{912}, and its 1-in-centrality is $8.59 \cdot 10^7$. The associated chained structure is $\{86, 47\}$-chained with initial vertex v_{541}. These large values of the bandwidth k are not surprising since bus networks are usually made up by several path networks.

In Figure 16, red and black dots indicate the nodes in the minimum broadcasting set and the minimum receiving set, respectively. Similarly as in the case of the airport data set, the vertices in the minimum broadcasting set and receiving set are mostly not well-connected bus stops. Indeed, all but two of them belong to suburban areas of Quartu Sant'Elena and the remaining two are non-central bus stops in Cagliari. This example shows that, even if nodes in the minimum broadcasting can reach every other node in the network at the minimum cost, they are not easily reachable from the rest of the network. A similar statement can be made for the minimum receiving set.

6. Conclusion

This paper elucidates the relation between directed graphs and spanning forests. We define minimal broadcasting and receiving sets, as well as out-central nodes and in-central nodes. These notions are useful, e.g., for studying communication and transportation networks, as well as for city planning. Several examples are discussed in the paper.

Aknowledgment

The authors would like to thank the referees for comments. CF and GR were partially supported by Fondazione di Sardegna, Progetto biennale bando 2021, "Computational Methods and Networks in Civil Engineering (COMANCHE)", and by the INdAM-GNCS 2023 project "Tecniche numeriche per lo studio dei problemi inversi e l'analisi delle reti complesse".

References

[1] Bapat, B. P., Graphs and Matrices, Springer, London, 2001.

[2] Borgatti, S. P., Centrality and network flow, Soc. Netw. 27 (2005) 55-71.

[3] Concas, A., Fenu, C., Reichel, L., Rodriguez, G., and Zhang, Y., Chained structure of directed graphs with applications to social and transportation networks, Appl. Netw. Sci. 7 (2022), art. 64.

[4] Concas, A., Reichel, L., Rodriguez, G., and Zhang, Y., Chained graphs and some applications, Appl. Netw. Sci. 6 (2021), art. 39.

[5] De la Cruz Cabrera, O., Jin, J., Noschese, S., and Reichel, L., Communication in complex networks, Appl. Numer. Math. 172 (2022) 186–205.

[6] Deo, N., Graph Theory with Applications to Engineering and Computer Science, Dover, Mineola, 2017.

[7] Estrada, E., The Structure of Complex Networks: Theory and Applications,Oxford University Press, Oxford, 2012.

[8] Fenu, C., and Higham, D. J., Estimating and increasing the structural stability of a network, SIAM J. Matrix Anal. Appl. 38 (2017) 343–360.

[9] Newman, M. E. J., Networks: An Introduction, Oxford University Press, Oxford, 2010.

[10] Noschese, S., and Reichel, L., Estimating and increasing the structural robustness of a network, Numer. Linear Algebra Appl. 29 (2022), art. e2418.