A branch-and-cut algorithm for the κ-connected Forest Star Problem

Alfie Plant

University of Edinburgh

The κ-connected Forest Star Problem seeks to find a minimum directed forest through a subset of vertices while ensuring κ-connectivity - that is, each vertex is a member of κ distinct, directed trees. This property relates to survivability, an important feature in the design of networks, particularly in telecommunications and renewable energy infrastructure. This paper explores two formulations of the problem as an integer linear program. The first extends naturally from the case where $\kappa = 1$, while the second is a reduced formulation that will be proven equivalent to its prior. Using a branch-and-cut algorithm, it will be shown that the κ-FSP can be solved consistently for cases up to 100 vertices and that the reduced formulation computationally outperforms the naive formulation in such instances.

1 Introduction

Consider a directed graph $\mathcal{G} = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ such that $\mathcal{V} = \{1, 2, \ldots, n\}$ and $\mathcal{R} = \{n+1, \ldots, n+|\mathcal{R}|\}$ denote the vertex and root sets and $\mathcal{A} = (\mathcal{V} \cup \mathcal{R}) \times \mathcal{V}$ is the arc set. Let $\mathcal{A}_{\mathcal{V}} = \mathcal{V} \times \mathcal{V}$ define $\mathcal{A}_r = (\mathcal{V} \cup \{r\}) \times \mathcal{V}$ for each $r \in \mathcal{R}$. A directed tree rooted at $r \in \mathcal{R}$ is a subgraph of $\mathcal{G}$ that contains exactly one path from r to at least one vertex $i \in \mathcal{V}$. A directed tree can also be referred to as an arborescence. A directed forest is a disjoint union of directed trees.

Given a positive integer $\kappa \leq |\mathcal{R}|$, non-negative tree cost c_{ij} associated to $(i,j) \in \mathcal{A}$ and non-negative assignment cost a_{ij} associated to $(i,j) \in \mathcal{A}_{\mathcal{V}}$, the aim of the κ-connected Forest Star Problem (κ-FSP) is to find the set of directed trees, each rooted at a different $r \in \mathcal{R}$, such that each vertex $i \in \mathcal{V}$ is either on κ trees, or is assigned to a vertex that is on κ trees, which minimises the total tree and assignment cost. An example solution to the κ-FSP is illustrated in Figure 1.

The κ-FSP is NP-hard because the Maximum Leaf Spanning Arborescence Problem (MLSAP), which is known to be NP-hard [1], is a particular case of it. The MLSAP determines an arborescence of $\mathcal{G}$ that maximises the number of vertices that are not on the arborescence, but are assigned to a vertex that is on the arborescence. The κ-FSP is equivalent to the MLSAP when is $|\mathcal{R}| = 1$, $a_{ij} = -1$ for $(i,j) \in \mathcal{A}$ and $c_{ij} = 0$ for $(i,j) \in \mathcal{A}_{\mathcal{V}}$. Similarly, the p-arborescence problem finds an arborescence containing p vertices, surrounded by localised clusters of the remaining vertices, typifying the design of wireless sensor networks [2].

There is no primal heuristic for the κ-FSP, but one has been derived for the Minimum Spanning Arborescence Problem (MSAP) [3, 4]. The MSAP is equivalent to the κ-FSP when $a_{ij} = c_{ij}$ for $(i,j) \in \mathcal{A}_{\mathcal{V}}$ and $|\mathcal{R}| = 1$. Extending this problem to involve assignment costs that are independent from tree costs, analogously to the MLSAP and P-ASP, can offer a more practical solution since it connects a larger number of customers to the network through local terminals. Another similar problem for which efficient algorithms exist is the Ring Star Problem (RSP), which locates a ring that plays the same role as the forest in the κ-FSP, by constituting the backbone of the network [5, 6].

The Forest Star Problem (FSP) is a baseline case of the κ-FSP for $\kappa = 1$. It differs from the problems that have been discussed by considering a graph which contains more than one root, a feature present in many applications. This problem provides the building blocks for the formulations to the κ-FSP. The motivation to extend the FSP to $\kappa > 1$ is to preserve flow to vertices, despite the failure of some of the arcs or roots. This has a direct application to survivable network design. Traditionally, networks have not been planned with the infrastructure to survive the failure of a single cable cut or pipeline burst. The forest-like structure of survivable networks ensure that customers or re-fuelling stations can still receive the resource that flows through the network, even if it experiences damage.

The aim of the paper is to present two different formulations for the κ-FSP as an integer linear program. The first formulation, referred to as the naive κ-FSP, takes its name from its role as a clear iteration of the FSP. The second formulation, known as the reduced κ-FSP, is not as explicitly defined, but manages to substantially decrease the dimensions of the problem. A branch-and-cut algorithm will be presented that overcomes the exponential number of constraints that are required to preserve connectivity in a solution by adding cuts dynamically using a separation problem similar to one applied in [6]. The fundamental result of this paper is to prove equivalence between the formulations for the κ-FSP. The limitation of the reduced formulation for the κ-FSP is that a solution does not provide a tree decomposition, which, in a practical application, would be essential to the design of the network. However, the details for an algorithm to derive a tree decomposition will be outlined. A flow optimisation model that yields the same decomposition and performs more efficiently than the algorithm will also be presented. To compare the efficiency of both formulations, the branch-and-cut algorithm will be implemented and it will be shown that for larger problem sizes, the reduced formulation can find an optimal solution almost 50 times faster than the naive formulation.

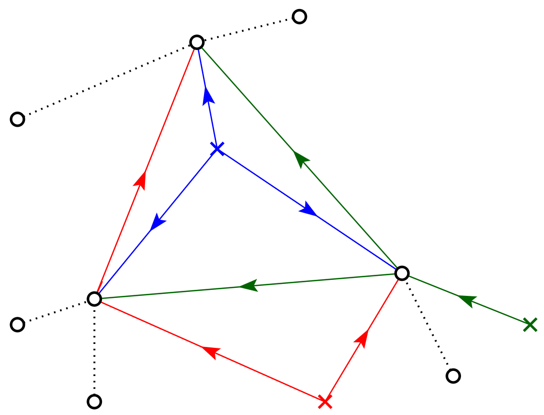
Figure 1: An example of a κ-FSP solution for $|\mathcal{V}| = 8, |\mathcal{R}| = 3, \kappa = 3$.

Definition 1 (Cut Set)**.** *Let $\mathcal{G} = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ be a directed graph. The cut sets of $\mathcal{G}$ associated to vertices $\mathcal{S} \subset \mathcal{V} \cup \mathcal{R}$ and arcs $\mathcal{T} \subseteq \mathcal{A}$ are defined as*

$$\delta^+(\mathcal{T}, \mathcal{S}) = \{(i,j) \in \mathcal{T} : i \notin \mathcal{S}, j \in \mathcal{S}\}$$
$$\text{and } \delta^-(\mathcal{T}, \mathcal{S}) = \{(i,j) \in \mathcal{T} : i \in \mathcal{S}, j \notin \mathcal{S}\}.$$

For simplicity, if $\mathcal{S} = \{i\}, \delta^+(\mathcal{T}, i)$ and $\delta^-(\mathcal{T}, i)$ are used instead of $\delta^+(\mathcal{T}, \{i\})$ and $\delta^-(\mathcal{T}, \{i\})$. Similarly, if $\mathcal{T} = \mathcal{A}$, $\delta^+(\mathcal{S})$ and $\delta^-(\mathcal{S})$ are used instead of $\delta^+(\mathcal{A}, \mathcal{S})$ and $\delta^-(\mathcal{A}, \mathcal{S})$[7].

Definition 2 (Minimum Cut)**.** *Given a directed graph $\mathcal{G} = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ together with some $s \in \mathcal{V} \cup \mathcal{R}$ and $t \in \mathcal{V} \cup \mathcal{R}$, the minimum cut of $\mathcal{G}$ is defined as*

$$\mathit{mincut}(\mathcal{G}, s, t) = (\Delta, \mathcal{C}^*, \mathcal{S}_0^*, \mathcal{S}_1^*).$$

The minimum cut is the minimum sum of capacities Δ of arcs in $\mathcal{C}^*$, which, when removed from $\mathcal{G}$, partitions $\mathcal{V} \cup \mathcal{R}$ into two subsets $\mathcal{S}_0^*$ and $\mathcal{S}_1^*$ such that $s \in \mathcal{S}_0^*$ and $t \in \mathcal{S}_1^*$. It follows that $\delta^-(\mathcal{S}_0^*) = \delta^+(\mathcal{S}_1^*) = \mathcal{C}^*$ [8].

Definition 3 (Directed Path)**.** *Let $\mathcal{G} = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ be a directed graph. A directed path $\mathcal{P}_{ij} = \{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ is a set of $k-1$ distinct arcs and corresponding sequence of k distinct vertices $(v_1, v_2, \ldots, v_k)$ such that $\alpha_m = (v_m, v_{m+1})$ for $m = 1, 2, \ldots, k-1$, $v_1 = i$ and $v_k = j$* [7].

Definition 4 (Flow)**.** *Let $\mathcal{G} = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ be a directed graph in which each arc $(i, j) \in \mathcal{A}$ has a non-negative capacity $\omega_{ij} \geq 0$. Given some $s \in \mathcal{V} \cup \mathcal{R}$ and $t \in \mathcal{S} \cup \mathcal{R}$, a flow through $\mathcal{G}$ is a mapping $\phi \to \mathbb{R}^+$ denoted ϕ_{ij} for some $(i, j) \in \mathcal{A}$, subject to the following constraints:*

1. ***Capacity constraint**:* $\phi_{ij} \leq \omega_{ij} \qquad \forall (i, j) \in \mathcal{A}$
2. ***Conservation of flow**:* $\sum_{(i,j) \in \delta^+(i)} \phi_{ij} = \sum_{(i,j) \in \delta^-(i)} \phi_{ij} \;\; \forall i \in \mathcal{V} \cup \mathcal{R} \setminus \{s, t\}$

Flow can be visualised as gas flowing through a pipeline. The volume of gas flowing through a pipe cannot exceed the volume of the pipe. Additionally, the volume of gas that enters a pipe should equal the volume of gas that leaves it. The value of a flow is defined by

$$|\phi| = \sum_{(s,j) \in \delta^-(s)} \phi_{sj} = \sum_{(i,t) \in \delta^+(t)} \phi_{it}.$$

The maximum flow problem is then to maximise $|\phi|$, meaning to route as much flow as possible from s to t [8].

Under the assumption that each arc in $\mathcal{G}$ has positive capacity, that is $\omega_{ij} > 0$ for each $(i, j) \in \mathcal{A}$, then if $|\phi| = 0$ for some $s \in \mathcal{V} \cup \mathcal{R}$, $t \in \mathcal{V} \cup \mathcal{R}$, then there does not exist a directed path from s to t. The following theorem is an important result in optimization theory that will be used throughout the paper.

Theorem 1 (Max-flow Min-cut)**.** *Given a directed graph $\mathcal{G} = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ with $\omega_{ij} \geq 0$ for $(i, j) \in \mathcal{A}$, together with some $s \in \mathcal{V} \cup \mathcal{R}$ and $t \in \mathcal{V} \cup \mathcal{R}$, the maximum flow in $\mathcal{G}$ is equal to the minimum cut of $\mathcal{G}$, that is $|\phi| = \Delta$* [9].

The remainder of this paper will be structured as follows: Section 2 describes the mathematical formulation of the FSP as well as the naive and reduced formulation for the κ-FSP. In section 3, a branch-and-cut algorithm that is used to solve these problems is outlined. A proof for the equivalence of the naive and reduced formulations is presented in section 4. This is followed by a set of computational experiments in section 5 and conclusions in section 6.

2 Mathematical Formulation

2.1 FSP

The forest star problem can be formulated as an integer linear program. Define binary decisions variables

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \in \mathcal{A} \text{ is in a tree} \\ 0 & \text{otherwise.} \end{cases}$$

Then, for each arc $(i,j) \in \mathcal{A}_{\mathcal{V}}$, define binary decision variables

$$y_{ij} = \begin{cases} 1 & \text{if vertex } i \in \mathcal{V} \text{ is assigned to vertex } j \in \mathcal{V} \\ 0 & \text{otherwise.} \end{cases}$$

If vertex $i \in \mathcal{V}$ is on the tree, it is then assigned to itself, that is $y_{ii} = 1$. The problem is then formulated as:

$$\begin{aligned}
\min \quad & \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \sum_{(i,j)\in\mathcal{A}_{\mathcal{V}}} a_{ij}y_{ij} && && (1)\\
\text{s.t.} \quad & \sum_{j\in\mathcal{V}} y_{ij} = 1 && \forall i \in \mathcal{V} && (2)\\
& y_{ii} \geq y_{ji} && \forall i \in \mathcal{V}, \forall j \in \mathcal{V} && (3)\\
& \sum_{(i,j)\in\delta^+(i)} x_{ji} = y_{ii} && \forall i \in \mathcal{V} && (4)\\
& \sum_{(i,j)\in\delta^+(i):j\neq i'} x_{ji} \geq x_{ii'} && \forall i \in \mathcal{V},\ \forall i' \in \mathcal{V} && (5)\\
& \sum_{(o,i)\in\delta^+(\mathcal{S})} x_{oi} \geq \sum_{j\in\mathcal{S}} y_{kj} && \forall \mathcal{S} \subseteq \mathcal{V},\ \forall k \in \mathcal{S} && (6)\\
& x_{ij} \in \{0,1\} && \forall (i,j) \in \mathcal{A} && (7)\\
& y_{ij} \in \{0,1\} && \forall (i,j) \in \mathcal{A}_{\mathcal{V}}. && (8)
\end{aligned}$$

The objective function (1) minimises the total tree and assignment cost. Constraints (2) ensure that each vertex is assigned to a vertex on the tree, or is assigned to itself, in which case it is on the tree. Constraints (3) guarantee that if vertex j is assigned to vertex i, then i must be on the tree. Constraints (4) make sure that only vertices on the tree have a predecessor on the tree. Constraints (5) ensure that if tree vertex i is a predecessor of i', that is $x_{ii'} = 1$, then it must have a predecessor that is not i'. To enforce connectivity, preventing solutions like the one illustrated in Figure 2, constraints (6) are required. These constraints are stronger than the classical constraints

$$\sum_{(o,i)\in\delta^+(\mathcal{S})} x_{oi} \geq y_{kk} \quad \forall \mathcal{S} \subseteq \mathcal{V} \quad \forall k \in \mathcal{S}. \qquad (9)$$

Suppose that $\mathcal{S}^* \subseteq \mathcal{V}$ violates connectivity and contains a vertex i with $y^*_{ii} = 0$. Then, (6) ensures that i would still produce a violation if i is assigned to some $j \in \mathcal{S}^*$. The lifted constraint is valid since (6) is equivalent to

$$\begin{aligned}
\sum_{(o,i)\in\delta^+(\mathcal{S})} x_{oi} &\geq \sum_{j\in\mathcal{V}} y_{kj} - \sum_{j\in\mathcal{V}\setminus\mathcal{S}} y_{kj} \\
&= 1 - \sum_{j\in\mathcal{V}\setminus\mathcal{S}} y_{kj} \qquad \text{by (2)}
\end{aligned}$$

for any $\mathcal{S} \subseteq \mathcal{V}$ and $k \in \mathcal{S}$. Then, if $y^*_{kk} = 1$, it follows that $\sum_{j \in \mathcal{V} \setminus \mathcal{S}} y_{kj} = 0$, and consequently (6) and (9) are equivalent. Conversely, if $y^*_{kk} = 0$, then (9) is satisfied trivially. Finally, (7) and (8) restrict decision variables to binary values.

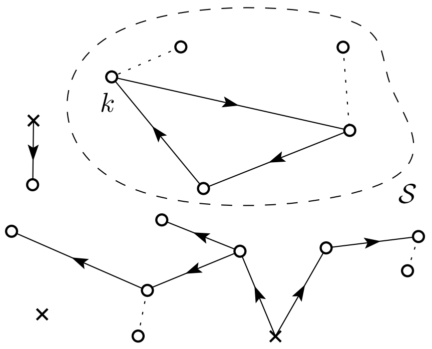

Figure 2: An illustration of a violation of (6) for subset $\mathcal{S}$ and vertex k.

2.1.1 Separation of Connectivity Constraints

To identify violations of connectivity constraint (6), let $\mathcal{G}^* = (\mathcal{V}^* \cup \mathcal{R}, \mathcal{A}^*)$ denote the support graph associated with a solution (x^*, y^*) of the linear relaxation (1)-(5), such that $\mathcal{V}^* = \{i \in \mathcal{V} : 0 < y^*_{ii} \leq 1\}$ and $\mathcal{A}^* = \{(i,j) \in \mathcal{A} : 0 < x^*_{ij} \leq 1\}$. A violation of constraints (6) is equivalent to finding some $\mathcal{S} \subset \mathcal{V}^*$ and $k \in \mathcal{S}$ such that

$$\sum_{(o,i) \in \delta^+(\mathcal{S})} x_{oi} + \sum_{j \in \mathcal{V} \setminus \mathcal{S}} y_{kj} \geq 1. \tag{10}$$

Clearly, if $\mathcal{G}^*$ is not connected, then each component $\mathcal{S} \subset \mathcal{V}^* \cup \mathcal{R}$ with $\mathcal{S} \cap \mathcal{R} = \emptyset$ is a violation for each $k \in \mathcal{S}$. If $\mathcal{G}^*$ is connected, finding a violation of the connectivity constraint can be reduced to a minimum cut problem, which is defined as follows. For some $k \in \mathcal{V}^*$, let $\mathcal{G}^{**}_k = (\mathcal{V}^{**} \cup \mathcal{R}, \mathcal{A}^{**})$ be a graph such that $\mathcal{V}^{**} = \mathcal{V}^* \cup \{s, t\}$ and $\mathcal{A}^{**} = \mathcal{A}^* \cup \mathcal{A}^s \cup \mathcal{A}^t$, where $\mathcal{A}^s = \{(s, r) : r \in \mathcal{R}\}$ and $\mathcal{A}^t = \{(i, t) : i \in \mathcal{V}^*\}$. Let capacity $\omega_{ij} = x^*_{ij}$ for $(i,j) \in \mathcal{A}^*$, $\omega_{sr} = 1$ for $(s,r) \in \mathcal{A}^s$ and $\omega_{it} = y^*_{ki}$ for $(i,t) \in \mathcal{A}^t$. If mincut$(\mathcal{G}^{**}, s, t)$ is such that $k \in \mathcal{S}^*_1$ and $\Delta < 1$, then $\mathcal{S} = \mathcal{S}^*_1 \setminus \{t\}$ defines the most violated connectivity constraint involving vertex k. This is because the dummy arcs $\mathcal{A}^t$ ensure that

$$\text{mincut}(\mathcal{G}^{**}_k, s, t)[\Delta] = \sum_{(o,i) \in \delta^+(\mathcal{S}^*_1)} x^*_{oi} + \sum_{j \in \mathcal{V}^* \setminus \mathcal{S}^*_1} y^*_{kj}.$$

2.2 Naive κ-FSP

The naive formulation for the κ-FSP can be viewed as $|\mathcal{R}|$ superimposed problems defined over $\mathcal{G}$. In a solution to the κ-FSP, each $r \in \mathcal{R}$ can be the root of a directed tree. The naive formulation incorporates this intuition by introducing decision variables associated to each $r \in \mathcal{R}$, allowing the problem to be solved in a way that separates the directed trees that are determined. For each $(i,j,r) \in \mathcal{A} \times \mathcal{R}$, define decision variables

$$x^r_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \in \mathcal{A} \text{ is in a tree rooted at } r \in \mathcal{R} \\ 0 & \text{otherwise.} \end{cases}$$

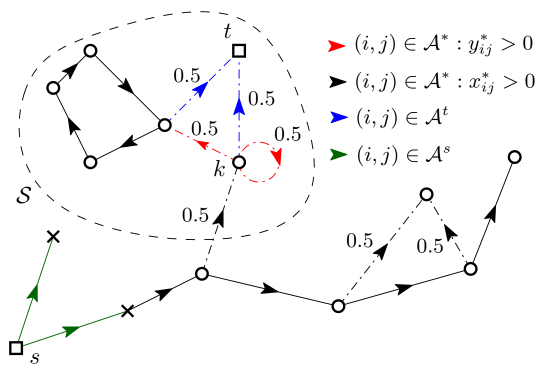
$(i,j) \in \mathcal{A}^* : y^*_{ij} > 0$
$(i,j) \in \mathcal{A}^* : x^*_{ij} > 0$
$(i,j) \in \mathcal{A}^t$
$(i,j) \in \mathcal{A}^s$
t
k
s
$\mathcal{S}$
0.5
0.5
0.5
0.5
0.5
0.5
0.5

Figure 3: An example of a violation of (6) in a solution to (1) - (5).

For each $(i, r) \in \mathcal{V} \times \mathcal{R}$, let

$$w_{ir} = \begin{cases} 1 & \text{if vertex } i \in \mathcal{V} \text{ is in a tree rooted at } r \in \mathcal{R} \\ 0 & \text{otherwise.} \end{cases}$$

Then, analagously to the FSP formulation, define

$$y_{ij} = \begin{cases} 1 & \text{if vertex } i \in \mathcal{V} \text{ is assigned to vertex } j \in \mathcal{V} \\ 0 & \text{otherwise} \end{cases}$$

for each $(i, j) \in \mathcal{A}_{\mathcal{V}}$. The naive κ-FSP is then formulated as:

$$\min \quad \sum_{(i,j,r)\in\mathcal{A}\times\mathcal{R}} c_{ij}x^r_{ij} + \sum_{(i,j)\in\mathcal{A}_{\mathcal{V}}} a_{ij}y_{ij} \tag{11}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{V}} y_{ij} = 1 \qquad \forall i \in \mathcal{V} \tag{12}$$

$$y_{ii} \geq y_{ji} \qquad \forall i \in \mathcal{V},\ \forall j \in \mathcal{V} \tag{13}$$

$$\sum_{r\in\mathcal{R}} w_{ir} = \kappa y_{ii} \qquad \forall i \in \mathcal{V} \tag{14}$$

$$\sum_{(i,j)\in\delta^+(\mathcal{A}_r,i):j\neq i'} x^r_{ji} \geq x^r_{ii'} \qquad \forall i \in \mathcal{V},\ \forall i' \in \mathcal{V},\ \forall r \in \mathcal{R} \tag{15}$$

$$\sum_{r\in\mathcal{R}} (x^r_{ij} + x^r_{ji}) \leq 1 \qquad \forall i \in \mathcal{V},\ \forall j \in \mathcal{V} \tag{16}$$

$$\sum_{(i,j)\in\delta^+(\mathcal{A}_r,i)} x^r_{ji} = w_{ir} \qquad \forall i \in \mathcal{V},\ \forall r \in \mathcal{R} \tag{17}$$

$$\sum_{(i,j)\in\delta^+(\mathcal{A}_r,\mathcal{S})} x^r_{ij} \geq w_{kr} \qquad \forall \mathcal{S} \subseteq \mathcal{V},\ \forall k \in \mathcal{S},\ \forall r \in \mathcal{R} \tag{18}$$

$$x^r_{ij} \in \{0, 1\} \qquad \forall (i, j) \in \mathcal{A} \times \mathcal{R} \tag{19}$$

$$w_{ir} \in \{0, 1\} \qquad \forall (i, r) \in \mathcal{V} \times \mathcal{R} \tag{20}$$

$$y_{ij} \in \{0, 1\} \qquad \forall (i, j) \in \mathcal{A}_{\mathcal{V}}. \tag{21}$$

Similarly to the objective function for the FSP, (11) minimises the total tree and assignment cost, but accounts for new decision variables x^r_{ij} for $(i, j, r) \in \mathcal{A} \times \mathcal{R}$. Constraints (12) and (13) are identical to constraints (2) and (3) respectively. Constraints (14) guarantee that if vertex i is on a tree, then it is on κ distinct trees. Constraints (15) are analogous to (5), but must be applied to each root separately. To ensure that each arc is on at most one tree in at most one direction, constraints (16) are required. Constraints (17) are imposed so that vertex i is on a tree rooted at r if and only if it has a predecessor j on a tree rooted at r. To force connectivity in a solution, constraints (18) make sure that if a subset of vertices $\mathcal{S} \subset \mathcal{V}$ contains a vertex k which is on tree r, then the incoming cut set of $\mathcal{S}$ will contain at least one arc that is on the tree rooted at r. Constraints (19)–(21) make sure that decision variables are binary.

2.2.1 Separation of Connectivity Constraints

The separation procedure for the naive κ-FSP does not require the introduction of dummy vertices or arcs. This is because a solution to the naive κ-FSP corresponds to arcs and vertices that are classified with respect to each $r \in \mathcal{R}$. As a result, violations of connectivity can be identified by dividing the solution of a linear relaxation into separate graphs associated to each root. Define $\mathcal{G}_r^* = (\mathcal{V}^* \cup \{r\}, \mathcal{A}_r^*)$ as the support graph of root r associated to a given solution (w^*, x^*, y^*) of the linear relaxation (11)–(17), where $\mathcal{V}^* = \{i \in \mathcal{V} : 0 < w_{ir}^* \leq 1\}$ and $\mathcal{A}_r^* = \{(i,j) \in \mathcal{A} : 0 < x_{ij}^{r*} \leq 1\}$. If $\mathcal{G}_r^*$ is not connected, each component $\mathcal{S}$ such that $r \notin \mathcal{S}$ is a violation of (18) for each $k \in \mathcal{S}$. Alternatively, if $\mathcal{G}_r^*$ is connected, consider

$$\text{mincut}(\mathcal{G}_r^*, r, k) = (\Delta, \mathcal{C}^*, \mathcal{S}_0^*, \mathcal{S}_1^*).$$

If $\Delta < w_{kr}^*$, then there is a violation of connectivity associated to $\mathcal{S} = \mathcal{S}_1^*$ and r for each $k \in \mathcal{S}$.

2.3 Reduced κ-FSP

The formulation for the reduced κ-FSP involves the same decision variables as the FSP, but does not introduce variables similarly to the naive κ-FSP. As a result, the dimensions of this formulation are considerably smaller. Table 1 highlights the scale of this difference.

	reduced	naive
variables	$\lvert\mathcal{A}\rvert + \lvert\mathcal{A}_{\mathcal{V}}\rvert + 2\lvert\mathcal{V}\rvert(\lvert\mathcal{V}\rvert - 1)$	$\lvert\mathcal{A}\rvert\lvert\mathcal{R}\rvert + \lvert\mathcal{A}_{\mathcal{V}}\rvert + 2\lvert\mathcal{V}\rvert(\lvert\mathcal{V}\rvert - 1) + \lvert\mathcal{V}\rvert^2\lvert\mathcal{R}\rvert$
constraints	$5\lvert\mathcal{V}\rvert^2 - 3\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}\rvert^2 + \lvert\mathcal{V}\rvert^2\lvert\mathcal{R}\rvert + \lvert\mathcal{V}\rvert$

Table 1: Standard form polyhedra dimensions for linear relaxation of κ-FSP formulations.

Analagously to the FSP formulation, define

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \in \mathcal{A} \text{ is in a tree} \\ 0 & \text{otherwise} \end{cases}$$

associated to each arc $(i,j) \in \mathcal{A}$ and

$$y_{ij} = \begin{cases} 1 & \text{if vertex } i \in \mathcal{V} \text{ is assigned to vertex } j \in \mathcal{V} \\ 0 & \text{otherwise} \end{cases}$$

for each arc in $(i,j) \in \mathcal{A}_{\mathcal{V}}$. The reduced κ-FSP is formulated as:

$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \sum_{(i,j)\in\mathcal{A}_{\mathcal{V}}} a_{ij}y_{ij} \tag{22}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{V}} y_{ij} = 1 \qquad \forall i \in \mathcal{V} \tag{23}$$

$$y_{ii} \geq y_{ji} + x_{ji} + x_{ij} \qquad \forall i \in \mathcal{V},\ \forall j \in \mathcal{V} \tag{24}$$

$$\sum_{(i,j)\in\delta^+(i)} x_{ji} = \kappa y_{ii} \qquad \forall i \in \mathcal{V} \tag{25}$$

$$\sum_{(i,j)\in\delta^+(i):j\neq i'} x_{ji} \geq x_{ii'} \qquad \forall i \in \mathcal{V}\ \forall i' \in \mathcal{V} \tag{26}$$

$$x_{ij} + x_{ji} \leq 1 \qquad \forall i \in \mathcal{V},\ \forall j \in \mathcal{V} \tag{27}$$

$$\sum_{(o,i)\in\delta^+(\mathcal{S})} x_{oi} \geq \sum_{j\in\mathcal{S}} y_{kj}(\kappa - |\mathcal{S}\cap\mathcal{R}|) \qquad \forall \mathcal{S} \subset \mathcal{V}\cup\mathcal{R} : |\mathcal{S}| > 2, \forall k \in \mathcal{S} \tag{28}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A} \tag{29}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}_{\mathcal{V}}. \tag{30}$$

The objective function (22) follows directly from (1) and constraints (23) and (26) are equivalent to (2) and (5) respectively. Constraints (25) are similar to (14), however since there is no distinction of which root an arc is assigned to, it guarantees that vertex i has κ predecessors if i is on a tree. Constraints (27) ensure that arc (i,j) and (j,i) are not used simultaneously for each $i \in \mathcal{V}$ and $j \in \mathcal{V}$. This restriction is not required for the FSP, since only one tree is being determined in the FSP and optimality satisfies (27). Constraints (28) play the role of ensuring connectivity in the solution. When $\mathcal{S}\cap\mathcal{R} = \emptyset$, (28) is analogous to (6), except we require κ arcs in the incoming cut set for $\mathcal{S}$ since the reduced κ-FSP preserves connectivity to κ roots simultaneously. When $\mathcal{S}\cap\mathcal{R} \neq \emptyset$, at least $\kappa - |\mathcal{S}\cap\mathcal{R}|$ arcs are required in the incoming cut set to guarantee connectivity. Similarly to the FSP, (28) are stronger than the classical constraints

$$\sum_{(o,i)\in\delta^+(\mathcal{S})} x_{oi} \geq y_{kk}(\kappa - |\mathcal{S}\cap\mathcal{R}|) \quad \forall \mathcal{S} \subset \mathcal{V}\cup\mathcal{R} : |\mathcal{S}| > 2, \forall k \in \mathcal{S} \tag{31}$$

The validity of (28) follows the same argument to the FSP formulation. For any $\mathcal{S} \subset \mathcal{V}\cup\mathcal{R}$ and $k \in \mathcal{S}$, (28) is equivalent to

$$\sum_{(o,i)\in\delta^+(\mathcal{S})} x_{oi} \geq (1 - \sum_{j\in\mathcal{V}\setminus\mathcal{S}} y_{kj})(\kappa - |\mathcal{S}\cap\mathcal{R}|). \tag{32}$$

For a given solution $(\bar{x},\bar{y})$ to the reduced κ-FSP, if $\bar{y}_{kk} = 1$ for some $k \in \mathcal{V}$, then (31) and (32) are equivalent and if $\bar{y}_{kk} = 0$, (31) is satisfied trivially.

Notice that (28) is only included for $\mathcal{S} \subset \mathcal{V}\cup\mathcal{R}$ such that $|\mathcal{S}| > 2$. Firstly, it should be noted that (28) holds trivially for any $\mathcal{S} \subseteq \mathcal{R}$. Then, if $\mathcal{S} = \{i\}$ for some $i \in \mathcal{V}$, (28) is equivalent to (25). Similarly, if $\mathcal{S} = \{r, i\}$ for some $r \in \mathcal{R}$ and $i \in \mathcal{V}$, (25) ensures that (28) holds. Constraints (24) have been tightened from (3) so that (28) holds when $\mathcal{S} = \{i, i'\}$ for some $i \in \mathcal{V}, i' \in \mathcal{V}$. Without loss of generality, suppose vertex i is assigned to i', that is, $y_{ii'} = 1$. Then, a combination of (24) and (25) make sure that (28) holds. Suppose that i is a predecessor to i'. Similarly, (24) and (25) ensure that connectivity is preserved. Finally, (29) and (30) enforce binary restrictions on decision variables.

2.3.1 Separation of Connectivity Constraints

Solving the separation problem for the reduced κ-FSP follows a similar process to the FSP. Define $\mathcal{G}^* = (\mathcal{V}^* \cup \mathcal{R}, \mathcal{A}^*)$ as a support graph associated to a solution $(\bar{x}, \bar{y})$ of the linear relaxation (22)–(27) such that $\mathcal{V}^* = \{i \in \mathcal{V} : 0 < \bar{y}_{ii} \leq 1\}$ and $\mathcal{A}^* = \{(i,j) \in \mathcal{A} : 0 < \bar{x}_{ij} \leq 1\}$. A violation of (28) is equivalent to finding $\mathcal{S} \subset \mathcal{V}^* \cup \mathcal{R}$ and $k \in \mathcal{S}$ such that

$$\sum_{(o,i)\in\delta^+(\mathcal{S})} \bar{x}_{oi} + |\mathcal{S} \cap \mathcal{R}| \geq \kappa \sum_{j\in\mathcal{S}} \bar{y}_{kj}. \tag{33}$$

If $\mathcal{G}^*$ is not connected and contains a component $\mathcal{S}$ such that $\mathcal{S} \cap \mathcal{R} = \emptyset$, then this violates the connectivity constraints for each $k \in \mathcal{S}$. If $\mathcal{G}^*$ is connected, violations of (28) can be identified using the following minimum cut problem. Define $\mathcal{G}^{**} = (\mathcal{V}^{**} \cup \mathcal{R}, \mathcal{A}^{**})$ with $\mathcal{V}^{**} = \mathcal{V}^* \cup \{s\}$ and $\mathcal{A}^{**} = \mathcal{A}^* \cup \mathcal{A}^s$, where $\mathcal{A}^s = \{(s,r) : r \in \mathcal{R}\}$. Let $\omega_{ij} = \bar{x}_{ij}$ for $(i,j) \in \mathcal{A}^*$ and $\omega_{sr} = 1$ for $(s,r) \in \mathcal{A}^s$. Since $\mathcal{A}^* \cap \mathcal{A}^s = \emptyset$ and $\mathcal{A}^* \cup \mathcal{A}^s = \mathcal{A}^{**}$, when considering cut sets over $\mathcal{G}^{**}$, the following holds:

$$\begin{aligned}\sum_{(o,i)\in\delta^+(\mathcal{S})} \bar{x}_{oi} &= \sum_{(o,i)\in\delta^+(\mathcal{A}^*,\mathcal{S})} \bar{x}_{oi} + \sum_{(o,i)\in\delta^+(\mathcal{A}^s,\mathcal{S})} \bar{x}_{oi} \\ &= \sum_{(o,i)\in\delta^+(\mathcal{A}^*,\mathcal{S})} \bar{x}_{oi} + |\mathcal{S} \cap \mathcal{R}|.\end{aligned}$$

It follows that mincut$(\mathcal{G}^{**}, s, k)[\Delta]$ is eqiuvalent to the left hand side of (33). As a result, when

$$\text{mincut}(\mathcal{G}^{**}, s, k)[\Delta] < \kappa \sum_{j\in\mathcal{S}_1^*} \bar{y}_{kj}$$

$\mathcal{S}_1^*$ and k produce a violation of (28).

3 Branch-and-cut Algorithm

In this section, the details to the branch-and-cut algorithm for finding an optimal solution of the FSP, naive κ-FSP and reduced κ-FSP will be outlined. For each formulation, the structure of the algorithm is the same, but the separation procedure varies. The details of these procedures that were discussed previously will be referenced throughout. The main drawback when solving these problems using a branch-and-bound algorithm is the exponential number of connectivity constraints (6), (18) and (28). To overcome this, these constraints are relaxed and added dynamically within the branch-and-cut algorithm, which is carried out as follows:

Step 1: Initialisation
Define the initial branch-and-bound subproblem as a linear relaxation of the formulation which excludes connectivity constraints and integer restrictions on decision variables.

Step 2: Subproblem Selection
Select a subproblem from the list of subproblems. Determine the LP solution. If the LP solution constitutes a new lower bound, update the lower bound. If the solution is integer feasible, go to Step 3a. If the solution is fractional, go to Step 3b.

Step 3a: Lazy Callback
Perform a connectivity check to find violated constraints. In the case of the FSP and naive κ-FSP this can be done by checking if the support graph associated to a specific root is connected. If it is not connected and there is a component that does not contain a root, add the violated cut and go to step 2. Otherwise, update the incumbent. If there are no remaining

nodes, terminate. Otherwise, go to step 2. A similar procedure is carried out for the reduced κ-FSP, however, a connectivity check is performed using the separation problem described in section 2.3.1. An illustration of this step is given in Figure 4.

Step 3b: User Callback
Initiate the separation procedure to find violated cuts. The details for the FSP are found in section 2.1.1, for the naive κ-FSP in section 2.2.1 and the reduced κ-FSP in section 2.3.1. If no cuts are found, go to step 4. Otherwise, add the violated cuts and return to step 2.

Step 4: Branching
Create two sub-problems by branching on a fractional decision variable. Go to step 2.

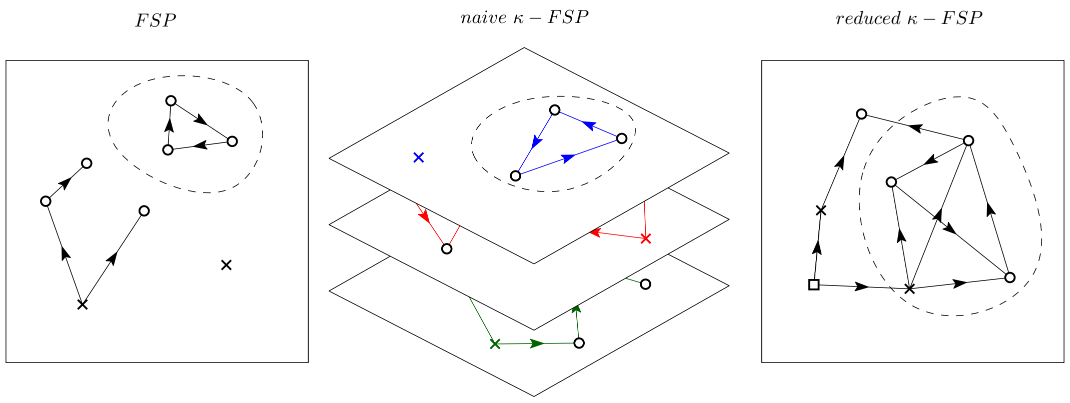

Figure 4: Examples of connectivity violations for integer feasible solutions to FSP and κ-FSP.

4 Equivalence of κ-FSP Formulations

In this section, equivalence of the reduced κ-FSP and naive κ-FSP will be derived for the case when $\kappa = |\mathcal{R}|$.

Proposition 1. *For any optimal solution (x^*, y^*, w^*) to the naive κ-FSP, there exists a mapping to a feasible solution $(\bar{x}, \bar{y})$ of the reduced κ-FSP.*

Proof. Let (x^*, y^*, w^*) be a solution to the naive κ-FSP and define

$$\bar{x}_{ij} = \sum_{r \in \mathcal{R}} x_{ij}^{r*} \qquad \forall (i,j) \in \mathcal{A}. \tag{34}$$

$$\text{and} \quad \bar{y}_{ij} = y_{ij}^* \qquad \forall (i,j) \in \mathcal{A}_{\mathcal{V}} \tag{35}$$

Constraints (16) ensure that $\bar{x}$ is binary and $\bar{y}$ is equivalent to y^*, therefore (29) and (30) are satisfied. Constraints (23) are identical to (12). Under this mapping, (26) and (27) are also equivalent to (15) and (16) respectively . A combination of (14) and (17) summed over $\mathcal{R}$ gives (25). For (24), assume for contradiction there is some $i \in \mathcal{V}, j \in \mathcal{V}$ such that $\bar{y}_{ii} < \bar{y}_{ji} + \bar{x}_{ji} + \bar{x}_{ij}$. Suppose that vertex i is not on a tree, that is $\bar{y}_{ii} = 0$. By (13), $\bar{y}_{ji} = 0$. Since, $\bar{y}_{ii} = 0$, (25) ensures that $\bar{x}_{ji} = 0$ and consequently (26) guarantees that $\bar{x}_{ij} = 0$, giving $0 < 0$, a contradiction. Conversely, suppose that vertex i is on the tree, that is $\bar{y}_{ii} = 1$. If $\bar{y}_{ji} = 1$, it follows that $\bar{y}_{jj} = 0$ by (12). Then, (25) and (26) make sure that $\bar{x}_{ij} = 0$ and $\bar{x}_{ji} = 0$, giving $1 < 1$. If $\bar{y}_{ij} = 0$, By (27), $\bar{x}_{ij} + \bar{x}_{ji} \leq 1$. In all scenarios, $\bar{y}_{ii} \geq \bar{y}_{ji} + \bar{x}_{ij} + \bar{x}_{ji}$. This contradicts the assumption, therefore (24) holds under the mapping.

The connectivity constraints (18) can be expressed as

$$\sum_{(i,j)\in\delta^+(\mathcal{A}_r,\mathcal{S})} x_{ij}^{r*} \geq w_{kr}^* - |\mathcal{S}\cap\{r\}| \quad \forall \mathcal{S}\subset\mathcal{V}^*\cup\mathcal{R},\ k\in\mathcal{S},\ r\in\mathcal{R}. \tag{36}$$

If $r\in\mathcal{S}$, it holds trivially and if $r\notin\mathcal{S}$, (36) follows from (18). Since $\delta^+(\mathcal{S})=\bigcup_{r\in\mathcal{R}}\delta^+(\mathcal{A}_r,\mathcal{S})$, summing (36) over $\mathcal{R}$ gives

$$\begin{aligned}
&\sum_{(i,j)\in\delta^+(\mathcal{S})} \bar{x}_{ij} \geq \sum_{r\in\mathcal{R}} w_{kr}^* - \sum_{r\in\mathcal{R}} \big|\mathcal{S}\cap\{r\}\big| && \text{by (34)}\\
\implies &\sum_{(i,j)\in\delta^+(\mathcal{S})} \bar{x}_{ij} \geq \kappa\bar{y}_{kk} - \sum_{r\in\mathcal{R}} \big|\mathcal{S}\cap\{r\}\big| && \text{by (14) and (35)}\\
\implies &\sum_{(i,j)\in\delta^+(\mathcal{S})} \bar{x}_{ij} \geq \kappa\bar{y}_{kk} - \big|\mathcal{S}\cap\mathcal{R}\big| \qquad \forall\mathcal{S}\subseteq\mathcal{V}\cup\mathcal{R}, k\in\mathcal{S}.
\end{aligned}$$

The last step holds because

$$\begin{aligned}
|\mathcal{S}\cap\mathcal{R}| &= \big|\mathcal{S}\cap(\{n+1\}\cup\{n+2\}\cup\cdots\cup\{n+|\mathcal{R}|\})\big|\\
&= \big|(\mathcal{S}\cap\{n+1\})\cup(\mathcal{S}\cap\{n+2\})\cup\cdots\cup(\mathcal{S}\cap\{n+|\mathcal{R}|\})\big|\\
&= |\mathcal{S}\cap\{n+1\}| + |\mathcal{S}\cap\{n+2\}| + \cdots + |\mathcal{S}\cap\{n+|\mathcal{R}|\}| \qquad \text{by disjointness}\\
&= \sum_{r\in\mathcal{R}} |\mathcal{S}\cap\{r\}|.
\end{aligned}$$

Since the classical constraints (31) hold under the mapping, (28) follows directly from this. Given that (23)–(30) are satisfied, it follows that the mapping described is a feasible solution to the reduced κ-FSP formulation. □

To help illustrate the next part of the equivalence proof, notice that a combination of constraints (14)–(18) ensure that there are κ disjoint paths from κ roots in $\mathcal{R}$ to each vertex $i\in\mathcal{V}$ for which $y_{ii}^*=1$. Intuitively speaking, the goal is to show that for a given solution $(\bar{x},\bar{y})$ to the reduced κ-FSP, there exists κ disjoint paths from κ roots in $\mathcal{R}$ to each tree vertex i such that $\bar{y}_{ii}=1$. The remaining part of this section will use this idea to build up to the next proposition.

Lemma 1. *For any solution $(\bar{x},\bar{y})$ to the reduced κ-FSP with $\kappa=|\mathcal{R}|$ there exists a path from each $r\in\mathcal{R}$ to k for all $k\in\mathcal{V}$ such that $\bar{y}_{kk}=1$.*

Proof. Let $\mathcal{G}^*=(\mathcal{V}^*\cup\mathcal{R},\mathcal{A}^*)$ denote the support graph associated to $(\bar{x},\bar{y})$. Assume for contradiction that there is some $k\in\mathcal{V}^*$ and non-empty $\mathcal{I}\subseteq\mathcal{R}$ such that there does not exist a path from each $r\in\mathcal{I}$ to k. Define $\mathcal{G}^{**}=(\mathcal{V}^*\cup\mathcal{R}\cup\{s\},\mathcal{A}^*\cup\{(s,r):r\in\mathcal{I}\})$ with $\omega_{sr}=1$ for each (s,r) such that $r\in\mathcal{I}$ and $\omega_{ij}=1$ for $(i,j)\in\mathcal{A}^*$. It follows that the maximum flow from s to k is 0. By the max-flow min-cut theorem,

$$\text{mincut}(\mathcal{G}^{**},s,k)=(\Delta=0,\mathcal{C}^*=\emptyset,\mathcal{S}_0^*,\mathcal{S}_1^*),$$

and consequently

$$\sum_{(i,j)\in\delta^+(\mathcal{S}_1^*)} x_{ij}^* = 0, \tag{37}$$

with $\mathcal{I}\subset\mathcal{S}_0^*$. Let $\mathcal{J}=\mathcal{R}\setminus\mathcal{I}$. By assumption, there exists a path from each $r\in\mathcal{J}$ to k, hence the flow from $r\in\mathcal{J}$ to k is greater than 0, and, by the optimality of the minimum cut problem, $\mathcal{J}\subset\mathcal{S}_1^*$. As a result, $|\mathcal{S}_1^*\cap\mathcal{R}|=\kappa-|\mathcal{I}|$ and, by (28),

$$\sum_{(i,j)\in\delta^+(\mathcal{S}_1^*)} x_{ij}^* \geq \kappa - |\mathcal{S}_1^*\cap\mathcal{R}| = \kappa-(\kappa-|\mathcal{I}|)=|\mathcal{I}|,$$

but by assumption $|\mathcal{I}| > 0$, contradicting (37). It follows that $|\mathcal{I}| = 0$, or equivalently, there exists a path from each root $r \in \mathcal{R}$ to k. $\square$

By Lemma 1, for any solution $(\bar{x}, \bar{y})$ to the κ-FSP, the support graph $\mathcal{G}^* = (\mathcal{V}^*, \mathcal{A}^*)$ contains a choice of paths $(\mathcal{P}^*_{(n+1,k)}, \mathcal{P}^*_{(n+2,k)}, \dots, \mathcal{P}^*_{(n+|\mathcal{R}|,k)})$ for each $k \in \mathcal{V}^*$. By (16), it is require that there exist a choice of paths that are arc disjoint.

Lemma 2. *For any solution $(\bar{x}, \bar{y})$ to the reduced κ-FSP with $\kappa = |\mathcal{R}|$ there exists a choice of paths $(\mathcal{P}_{(n+1,k)}, \mathcal{P}_{(n+2,k)}, \dots, \mathcal{P}_{(n+|\mathcal{R}|,k)})$ such that $\mathcal{P}_{(n+1,k)} \cap \mathcal{P}_{(n+2,k)} \cap \dots \cap \mathcal{P}_{(n+|\mathcal{R}|,k)} = \emptyset$ for each $k \in \mathcal{V}$ such that $\bar{y}_{kk} = 1$.*

Proof. Let $\mathcal{G}^* = (\mathcal{V} \cup \mathcal{R}, \mathcal{A})$ denote the support graph associated to $(\bar{x}, \bar{y})$ Assume for contradiction that there is some $k \in \mathcal{V}^*$ and non-empty $\mathcal{I} \subseteq \mathcal{R}$ such that for every choice of paths, $\bigcap_{r \in \mathcal{I}} \mathcal{P}^*_{rk} \subseteq \mathrm{A}$ with $\mathrm{A} \neq \emptyset$. Define $\mathcal{G}^{**} = (\mathcal{V}^* \cup \mathcal{R} \cup \{s\}, \mathcal{A}^* \cup \{(s,r) : r \in \mathcal{I}\})$ with $\omega_{sr} = 1$ for each (s,r) with $r \in \mathcal{I}$ and $\omega_{ij} = 1$ for $(i,j) \in \mathcal{A}^*$. In any choice of paths, each $\mathcal{P}^*_{rk}$ for which $r \in \mathcal{I}$ shares at least one arc with another path and it follows that the maximum flow from s to k is less than $|\mathcal{I}|$. By the max-flow min-cut theorem

$$\text{mincut}(\mathcal{G}^{**}, s, k) = (\Delta, \mathcal{C}^*, \mathcal{S}^*_0, \mathcal{S}^*_1)$$

where $\Delta < |\mathcal{I}|$ and, as a result,

$$\sum_{(i,j) \in \delta^+(\mathcal{S}^*_1)} \bar{x}_{ij} < |\mathcal{I}|. \tag{38}$$

Note that A contains the only arcs in $\mathcal{G}^{**}$ that, when cut from the graph, separate more than one root from k. It follows that $\mathcal{C}^* \subseteq \mathrm{A}^*$ by the optimality of the minimum cut problem. Since $\mathcal{A}_{\mathcal{D}} \cap \mathrm{A} = \emptyset, \mathcal{I} \subset \mathcal{S}^*_0$. Let $\mathcal{J} = \mathcal{R} \setminus \mathcal{I}$. By assumption, the paths from $r \in \mathcal{J}$ to k are disjoint, therefore it is possible to find $\mathcal{P}^*_{rk}$ for $r \in \mathcal{J}$ such that $\mathcal{P}^*_{rk} \cap \mathrm{A} = \emptyset$, and it follows that $\mathcal{J} \subset \mathcal{S}^*_1$. Then, (28) ensures that,

$$\sum_{(i,j) \in \delta^+(\mathcal{S}^*_1)} \bar{x}_{ij} \geq \kappa - |\mathcal{S}^*_1 \cap \mathcal{R}| = \kappa - (\kappa - |\mathcal{I}|) = |\mathcal{I}|,$$

contradicting (38) and there exists a choice of paths $(\mathcal{P}^*_{(n+1,k)}, \mathcal{P}^*_{(n+2,k)}, \dots, \mathcal{P}^*_{(n+|\mathcal{R}|,k)})$ such that $\mathcal{P}^*_{(n+1,k)} \cap \mathcal{P}^*_{(n+2,k)} \cap \dots \cap \mathcal{P}^*_{(n+|\mathcal{R}|,k)} = \emptyset$ $\square$

By Lemma 2, there exists a selection $(\mathcal{P}^*_{(n+1,k)}, \mathcal{P}^*_{(n+2,k)}, \dots, \mathcal{P}^*_{(n+|\mathcal{R}|,k)})$ for each $k \in \mathcal{V}^*$ and the assignment

$$x^{r*}_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \in \mathcal{P}^*_{rk} \\ 0 & \text{otherwise.} \end{cases}$$

could be made. However, to ensure that (16) is satisfied, the assignment given to one path cannot contradict another, that is,

$$\sum_{r \in \mathcal{R}} (x^{r*}_{ij} + x^{r*}_{ji}) \leq 1 \quad \forall (i,j) \in \mathcal{A}^*.$$

Definition 5 (Chokepoint Set)**.** *Let the support graph $\mathcal{G}^* = (\mathcal{V}^* \cup \mathcal{R}, \mathcal{A}^*)$ associated to a solution $(\bar{x}, \bar{y})$ of the κ-FSP be given. Define $\mathcal{C} \subset \mathcal{A}^*$ as a chokepoint set if there exists $k \in \mathcal{V}^*$ and $\mathcal{D} \subseteq \mathcal{R}$ with $|\mathcal{D}| = |\mathcal{C}| \leq \kappa$ that corresponds to $\mathcal{G}^{**} = (\mathcal{V}^* \cup \mathcal{R} \cup \{s\}, \mathcal{A}^* \cup \mathcal{A}_{\mathcal{D}})$ with $\mathcal{A}_{\mathcal{D}} = \{(s,r) : r \in \mathcal{D}\}$ and $\omega_{ij} = 1$ for $(i,j) \in \mathcal{A}^* \cup \mathcal{A}_{\mathcal{D}}$ such that*

$$\textit{mincut}(\mathcal{G}^{**}, s, k) = (\Delta = |\mathcal{C}|, \mathcal{C}^* = \mathcal{C}, \mathcal{S}^*_0, \mathcal{S}^*_1).$$

A chokepoint set $\mathcal{C}$ corresponding to $k \in \mathcal{V}^*$ and $\mathcal{D} \subseteq \mathcal{R}$ represents an instance when the path $\mathcal{P}^*_{rk}$ chosen from each $r \in \mathcal{D}$ to k must contain a corresponding arc in $\mathcal{C}$. To guarantee that all paths can be assigned without any contradictions, it must be shown that a chokepoint set $\mathcal{C}$ associated to $k \in \mathcal{V}^*$ and $\mathcal{D} \subseteq \mathcal{R}$ cannot be a chokepoint set associated to another vertex and another set of roots.

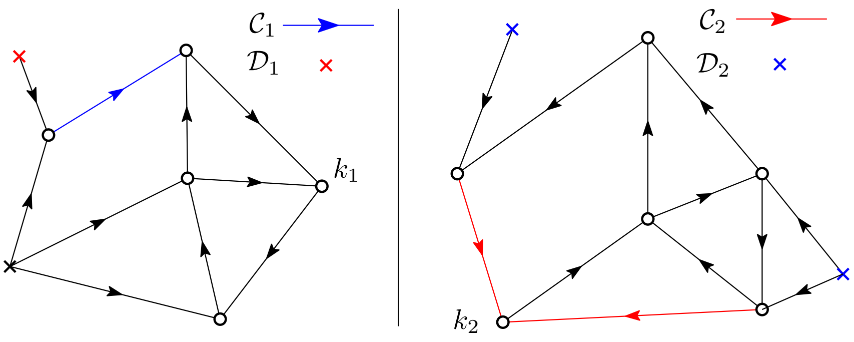

Figure 5: Chokepoint set examples $\mathcal{C}_1$ (left) and $\mathcal{C}_2$ (right) with $|\mathcal{C}_1| = 1$, $|\mathcal{C}_2| = 2$.

Lemma 3. *For any solution $(\bar{x}, \bar{y})$ of the reduced κ-FSP with $\kappa = |\mathcal{R}|$ and associated support graph $\mathcal{G}^* = (\mathcal{V}^* \cup \mathcal{R}, \mathcal{A}^*)$, there does not exist a chokepoint set $\bar{\mathcal{C}}$ associated to $k \in \mathcal{V}^*$, $\mathcal{D} \subset \mathcal{R}$ and $l \in \mathcal{V}^*$, $\mathcal{D}' \subset \mathcal{R}$ with $k \neq l$, $\mathcal{D} \cap \mathcal{D}' = \emptyset$.*

Proof. Assume for contradiction that such a chokepoint set $\bar{\mathcal{C}}$ exists. By definition of a chokepoint set, $\mathcal{G}^{**}_0 = (\mathcal{V}^* \cup \mathcal{R} \cup \{s\}, \mathcal{A}^* \cup \mathcal{A}_{\mathcal{D}})$ and $\mathcal{G}^{**}_1 = \left(\mathcal{V}^* \cup \mathcal{R} \cup \{s\}, \mathcal{A}^* \cup \mathcal{A}_{\mathcal{D}'}\right)$ can be constructed such that

$$\begin{aligned} &\operatorname{mincut}(\mathcal{G}^{**}_0, s, k) = (\Delta = |\mathcal{C}|, \mathcal{C}, \mathcal{S}^*_0, \mathcal{S}^*_1) \\ \textit{and}\quad &\operatorname{mincut}(\mathcal{G}^{**}_1, s, k) = (\Delta = |\mathcal{C}|, \mathcal{C}, \mathcal{S}'^*_0, \mathcal{S}'^*_1). \end{aligned}$$

Since $\mathcal{A}_{\mathcal{D}} \cap \mathcal{C} = \emptyset$ and $\mathcal{A}_{\mathcal{D}'} \cap \mathcal{C} = \emptyset$, it follows that $\mathcal{D} \subset \mathcal{S}^*_0$ and $\mathcal{D}' \subset \mathcal{S}'^*_0$. Let $\mathcal{J} = \mathcal{R} \setminus \mathcal{D}$ and $\mathcal{J}' = \mathcal{R} \setminus \mathcal{D}'$. By assumption, there exists a path $\mathcal{P}^*_{rk}$ for each $r \in \mathcal{J}$ such that $\mathcal{P}^*_{rk} \cap \mathcal{C} = \emptyset$, therefore $\mathcal{J} \subset \mathcal{S}^*_1$, otherwise the optimality of the minimum cut would not hold. By a similar argument, $\mathcal{J}' \subset \mathcal{S}'^*_1$. Let $\mathcal{S} = \mathcal{S}^*_1 \cap \mathcal{S}'^*_1$. Then, $|\mathcal{S} \cap \mathcal{R}| = \kappa - (|\mathcal{D}| + |\mathcal{D}'|) = \kappa - 2|\mathcal{C}|$. Since

$$\sum_{(i,j) \in \delta^+(\mathcal{S}^*_1)} \bar{x}_{ij} = |\mathcal{C}| \text{ and } \sum_{(i,j) \in \delta^+(\mathcal{S}'^*_1)} \bar{x}_{ij} = |\mathcal{C}|,$$

it follows that

$$\sum_{(i,j) \in \delta^+(\mathcal{S})} \bar{x}_{ij} \leq |\mathcal{C}|, \tag{39}$$

but by constraint (28)

$$\sum_{(i,j) \in \delta^+(\mathcal{S})} x^*_{ij} \geq \kappa - (\kappa - 2|\mathcal{C}|) = 2|\mathcal{C}|$$

which contradicts (39), since $|\mathcal{C}| > 0$ by assumption. Hence $\bar{\mathcal{C}}$ cannot exist. □

Lemmas 1, 2 and 3 guarantee that when a solution is found for the reduced κ-FSP with $\kappa = |\mathcal{R}|$, an algorithm can be implemented to assign each arc in the solution set to a tree rooted at some $r \in \mathcal{R}$. The pseudo-code for this process is presented in Algorithm 1.

Steps **1**-**3** initialise the solution (x^*, y^*, z^*). Then, steps **4**-**7** assign root arcs, which would be identified as chokepoints at a later stage, however it is more efficient to deal with them now since their assignment is trivial.

The role of steps **8**-**29** is to find all chokepoints in $\mathcal{G}^*$. Part of this process involves classifying chokepoint sets. Each arc in a chokepoint set $\mathcal{C}$ that is associated to $k \in \mathcal{V}^*$ and $\mathcal{D} \subseteq \mathcal{R}$ is necessary to ensuring that there is a path from each $r \in \mathcal{D}$ to k. When $|\mathcal{C}| > 1$, a mapping from each $(i, j) \in \mathcal{C}$ to each $r \in \mathcal{D}$ must be determined. Step **11** loops through all the relevant sets of arcs, but this is done in order of size, from smallest to largest. Consequently, the most critical chokepoint sets are identified first. Steps **17**-**20** carry out a classification when no arcs in $\mathcal{C}$ have previously been classified. When larger chokepoint sets are found, they might contain a smaller chokepoint set that has already been classified. When this is the case, steps **21**-**25** classify the arcs in the set correctly. If all arcs in a chokepoint set have been classified, then there is still information that must be saved. This is because a chokepoint arc can be essential to ensuring there is a path from some $r \in \mathcal{R}$ to more than one vertex of $\mathcal{G}^*$. Steps **26**-**29** carry out this procedure.

Once all chokepoints have been identified, paths can be assigned. The algorithm DFS used in step **31** finds all paths from any root $r \in \mathcal{R}$ to some $k \in \mathcal{V}^*$. The pseudo-code for this algorithm can be found in [8]. Steps **30**-**40** loop through each vertex $k \in \mathcal{V}^*$ randomly. In steps **31**-**35** each path is be checked for chokepoints. If k is not already on a tree rooted at r and the path contains all chokepoints required on a path from r to k represented by $\mathcal{C}_{rk}$ and no arcs on the path have been assigned to some other root, then this is a chokepoint path and is assigned. In steps **36**-**40**, the remaining paths can be assigned randomly. A path from r to k can be assigned if k is not already on a tree rooted at r and no arcs on the path have been assigned to some other root. Once all assignments have been made, Algorithm 1 returns the solution (x^*, y^*, w^*).

Proposition 2. *For any optimal solution $(\bar{x}, \bar{y})$ of the reduced κ-FSP with $\kappa = |\mathcal{R}|$, there exists a mapping to a feasible solution of the naive κ-FSP.*

Proof. Let a solution $(\bar{x}, \bar{y})$ to the reduced κ-FSP be given and define

$$(x^*, y^*, w^*) = \text{TREE\_DECOMPOSITION}(\bar{x}, \bar{y}).$$

Clearly, (19)-(20) hold because TREE_DECOMPOSITION only produces binary values and (12) holds since it is equivalent to (23). Constraints (13) hold for y^* since (24) is a tighter inequality. By Lemma 1, there exists a path from each root $r \in \mathcal{R}$ to each vertex $i \in \mathcal{V}$ such that $y^*_{ii} = 1$. Consequently, the mapping given by TREE_DECOMPOSITION guarantees a directed path from each root $r \in \mathcal{R}$ to each tree vertex. All vertices on a directed path are distinct, therefore (15) holds. Similarly, this ensures that if vertex i is on a tree, then it is on κ distinct trees, hence (14) and (17) are also satisfied. By Lemmas 2 and 3, the paths assigned to each tree vertex are disjoint, and (27) guarantees that individual arcs are not used in both directions. It follows that the mapping given by TREE_DECOMPOSITION meets the requirements of (16).

The final constraints that the mapping must satisfy are (18). Let $\mathcal{S} \subset \mathcal{V}$, $k \in \mathcal{S}$ and $r \in \mathcal{R}$ be given. If $w^*_{kr} = 0$, (18) holds trivially. Suppose $w^*_{kr} = 1$, that is k is on a tree rooted at r. By (17), k has some predecessor $j \in \mathcal{V}$, that is also on a tree rooted at r. If $j \notin \mathcal{S}$, it follows that

$$\sum_{(i,j) \in \delta^+(\mathcal{A}_r, \mathcal{S})} x^*_{ij} \geq 1 \tag{40}$$

and (18) is satisfied. If $j \in \mathcal{S}$, then TREE_DECOMPOSITION guarantees that j has some predecessor that is on a tree rooted at r. This process can be repeated until a vertex is found that is not in $\mathcal{S}$. By assumption, $r \notin \mathcal{S}$, therefore (18) will eventually hold. Since (12)–(18) are satisfied under the mapping, it follows that (x^*, y^*, w^*) is a solution to the naive κ-FSP. □

Algorithm 1: TREE_DECOMPOSITION

```
Input: optimal solution $(\bar{x}, \bar{y})$ to (22)–(28) and support graph $\mathcal{G}^* = (\mathcal{V}^* \cup \mathcal{R}, \mathcal{A}^*)$
$y^*_{ij} \leftarrow \bar{y}_{ij} \quad \forall (i,j) \in \mathcal{A}_\mathcal{V}$;
$x^{r*}_{ij} \leftarrow 0 \quad \forall (i,j,r) \in \mathcal{A} \times \mathcal{R}$;
$w^*_{ir} \leftarrow 0 \quad \forall (i,r) \in \mathcal{V} \times \mathcal{R}$;
$\bar{\mathcal{R}} \leftarrow \{(r,j) \in \mathcal{A}^* : r \in \mathcal{R}\}$;
foreach $(r,j) \in \bar{\mathcal{R}}$ do
    $x^{*r}_{rj} \leftarrow 1$
    $w^*_{jr} \leftarrow 1$
$\bar{\mathcal{A}} \leftarrow \emptyset$;
foreach $k \in \mathcal{V}^*$ do
    $\mathcal{C}_{rk} \leftarrow \emptyset \quad \forall r \in \mathcal{R}$;
    foreach $\mathcal{C} \subset \mathcal{A}^* \setminus \bar{\mathcal{R}} : 1 \leq |\mathcal{C}| \leq \kappa$ do
        foreach $\mathcal{D} \subseteq \mathcal{R} : |\mathcal{D}| = |\mathcal{C}|$ do
            $\mathcal{A}^{**} \leftarrow (\mathcal{A}^* \cup \mathcal{A}_\mathcal{D}) \setminus \mathcal{C}$;
            $\mathcal{V}^{**} \leftarrow \mathcal{V} \cup \mathcal{R} \cup \{s\}$;
            if $mincut(\mathcal{G}^{**} = (\mathcal{V}^{**}, \mathcal{A}^{**}), s, k)[\Delta] = 0$ then
                $\bar{\mathcal{A}} \leftarrow \bar{\mathcal{A}} \cup \mathcal{C}$;
                if $|\mathcal{C} \cap \bar{\mathcal{A}}| = 0$ then
                    foreach $(i,j) \in \mathcal{C}, r \in \mathcal{D}$ do
                        $x^{r*}_{ij} \leftarrow 1$;
                        $\mathcal{C}_{rk} \leftarrow \mathcal{C}_{rk} \cup \{(i,j)\}$
                if $0 < |\mathcal{C} \cap \bar{\mathcal{A}}| < |\mathcal{C}|$ then
                    $\mathcal{D}' = \{r \in \mathcal{D} : x^{r*}_{ij} = 0 \ \forall (i,j) \in \mathcal{C}\}$;
                    foreach $(i,j) \in \mathcal{C} \setminus \mathcal{C} \cap \bar{\mathcal{A}}, \ r \in \mathcal{D}'$ do
                        $x^{r*}_{ij} \leftarrow 1$
                        $\mathcal{C}_{rk} \leftarrow \mathcal{C}_{rk} \cup \{(i,j)\}$
                    if $|\mathcal{C} \cap \bar{\mathcal{A}}| = |\mathcal{C}|$ then
                        foreach $(i,j) \in \mathcal{C}$ do
                            $\bar{r} \leftarrow r \in \mathcal{R} : x^{r*}_{ij} = 1$
                            $\mathcal{C}_{\bar{r}k} \leftarrow \mathcal{C}_{\bar{r}k} \cup \{(i,j)\}$
foreach $k \in \mathcal{V}^*$ do
    foreach $\mathcal{P}_{rk} \in$ DFS$(\mathcal{G}^*, k)$ do
        if $w^*_{kr} = 0$ and $\mathcal{P}_{rk} \cap \bar{\mathcal{A}} = \mathcal{C}_{rk}$ and $x^{u*}_{ij} = 0 \ \forall (i,j) \in \mathcal{P}_{rk}, u \in \mathcal{R} \setminus \{r\}$ then
            foreach $(i,j) \in \mathcal{P}_{rk}$ do
                $x^{r*}_{ij} \leftarrow 1$;
                $w^*_{jr} \leftarrow 1$
    foreach $\mathcal{P}_{rk} \in$ DFS$(\mathcal{G}^*, k)$ do
        if $w^*_{kr} = 0$ and $x^{u*}_{ij} = 0 \ \forall (i,j) \in \mathcal{P}_{rk}, u \in \mathcal{R} \setminus \{r\}$ then
            foreach $(i,j) \in \mathcal{P}_{rk}$ do
                $x^{r*}_{ij} \leftarrow 1$;
                $w^*_{jr} \leftarrow 1$
Output: $(x^*, y^*, w^*)$
```

Theorem 2. *When $\kappa = |\mathcal{R}|$, the naive formulation of the κ-FSP is equivalent to the reduced formulation of the κ-FSP*

Proof. The result follows directly from Propositions 1 and 2. $\square$

Computational Complexity of TREE_DECOMPOSITION

A drawback of this algorithm is the computational complexity associated to steps **8-29**. To find all chokepoints, the maximum number of minimum cuts that could be taken is

$$\sum_{i=1}^{\kappa} \binom{|\mathcal{A}^*|}{i} \binom{\kappa}{i} |\mathcal{V}^*|.$$

Consequently, the algorithm has $\mathcal{O}(n!)$ which becomes an issue for larger problems that are likely to arise in practical applications. However, since we have proved the result of equivalence, we can use optimization to derive a solution with assigned paths for each $i \in \mathcal{V}^*$.

Given a solution (x^*, y^*) to the reduced κ-FSP, introduce a dummy vertex s and a set of arcs $\mathcal{A}_\mathcal{D} = \{(s, r) : r \in \mathcal{R}\}$. Then we can derive the following flow colouring model. Let $\phi_{ij}^r \in \mathbb{R}^+$ denote the number of units of flow of commodity $r \in \mathcal{R}$ routed from a vertex $i \in \mathcal{V}^*$ to a vertex $j \in \mathcal{V}^* \cup \{s\}$. Therefore, if each vertex included in a tree rooted at $r \in \mathcal{R}$ generates one unit of flow commodity r, then the constraints

$$w_{jr} + \sum_{i \in \mathcal{V}: i \neq j} \phi_{ij}^r = \sum_{i \in \mathcal{V}^* \cup \mathcal{R}: i \neq j} \phi_{ji}^r \qquad \forall j \in \mathcal{V}^*,\ r \in \mathcal{R} \tag{41}$$

$$\phi_{rs}^r = \sum_{i \in \mathcal{V}^*} w_{ir} \qquad \forall r \in \mathcal{R} \tag{42}$$

$$\phi_{ij}^r \leq |\mathcal{V}^*| x_{ji}^r \qquad \forall (i,j) \in \mathcal{A}^*, r \in \mathcal{R} \tag{43}$$

$$\sum_{r \in \mathcal{R}} w_{jr} = \kappa y_{jj}^* \qquad \forall j \in \mathcal{V}^* \tag{44}$$

$$\sum_{r \in \mathcal{R}} x_{ij}^r = x_{ij}^* \qquad \forall (i,j) \in \mathcal{A}^* \tag{45}$$

$$\sum_{j \in \mathcal{V}^* \cup \mathcal{R}} x_{ji}^r = w_{ir} \qquad \forall i \in \mathcal{V}^*, r \in \mathcal{R} \tag{46}$$

$$x_{ij}^r \leq \sum_{k \in \mathcal{V}^* \cup \mathcal{R}: k \neq j} x_{ki}^r \qquad \forall i \in \mathcal{V}^*, j \in \mathcal{V}^*, r \in \mathcal{R} \tag{47}$$

enforce a solution (x^*, w^*, ϕ^*) that coincides with the extended κ-FSP formulation. Constraints (41) guarantee conservation of flow. Constraints (42) make sure that the flow from each root r to the dummy s is equal to the number of vertices on tree r. Constraints (45) are in place so that each arc is associated to one tree and constraints (43) ensure that flow only exists on arcs that have been built, fulfilling the capacity constraint. Constraints (44) ensure (14) is satisfied, i.e. that each tree vertex j is on κ distinct trees. Constraints (46) meet the requirements of (17), that is, if a vertex i is on a tree associated to r, then it will have a predecessor on that tree. Finally, constraints (47) follow directly from (15). The formulation does not require an objective function because any feasible solution corresponds to an optimal solution of the extended formulation. Note that a combination of (41)–(43) preserve connectivity in the solution.

5 Computational Experiments

The branch-and-cut algorithm described in Section 3 was implemented with the DOcplex Python API. The performance of the naive κ-FSP and reduced κ-FSP was tested on five different seeds across 15 instances of data, including $|\mathcal{V}| = 20, 40, 60, 80, 100$ and $\kappa = 2, 4, 6$ with $\kappa = |\mathcal{R}|$. Table 2 and 3 outline the results for each run and Table 4 shows averages across these runs for comparison. The column headings are defined as follows:

$\boldsymbol{|\mathcal{V}|}$:	Number of vertices.
$\boldsymbol{\kappa}$:	The number of roots and the parameter for the κ-FSPS
seed :	Seed used to generate data.
build :	Total build time of the model (*secs.cecs*).
root :	Relative MIP gap at root node of branch-and-cut. If no nodes are explored, the value is the same as gap. (%)
nodes :	The number of nodes explored in branch-and-cut.
cut time :	The time spent making cuts in *Step 3a* and *Step 3b* of branch-and-cut. (*mins:secs.csecs*)
cuts :	The total number of cuts added within branch-and-cut.
gap :	Relative MIP gap after solver has been terminated. (%)
solve :	Time taken to solve model - note that the solver is terminated after one hour.
flow :	Time taken to produce solution using flow model (41)–(47).
alg :	Time taken to implement TREE_DECOMPOSITION.
opt. :	Total number of runs where an optimal solution was found.

Table 2: Naive κ-FSP results.

$|\mathcal{V}|$	κ	seed	build	root	nodes	cuts	cut time	gap	solve
		1	0.05	1.06	1	110	00.18	0.00	00.55
		2	0.06	0.67	9	266	00.27	0.00	00.95
20	2	3	0.05	0.00	0	165	00.09	0.00	00.25
		4	0.05	0.00	0	203	00.11	0.00	00.34
		5	0.13	1.09	16	194	00.32	0.00	00.93
		1	0.09	5.94	152	823	03.48	0.00	08.23
		2	0.09	3.37	18	246	01.14	0.00	03.92
20	4	3	0.09	7.77	153	932	03.09	0.00	06.96
		4	0.15	6.60	115	780	02.54	0.00	05.95
		5	0.09	7.55	273	811	04.54	0.00	09.38
		1	0.13	13.36	361	1060	08.81	0.00	17.83
		2	0.24	0.00	0	148	00.09	0.00	00.56
20	6	3	0.13	13.05	108	1129	04.18	0.00	11.56
		4	0.13	7.34	34	613	01.78	0.00	05.30
		5	0.19	2.81	11	332	01.10	0.00	06.16
		1	0.38	1.59	42	1119	03.13	0.00	06.79
		2	0.44	0.01	0	807	00.93	0.01	02.06
40	2	3	0.36	0.00	0	832	01.05	0.00	03.56
		4	0.42	2.39	40	758	02.25	0.00	04.36
		5	0.36	1.03	26	1047	02.01	0.00	04.60
		1	0.76	45.04	2409	52262	06:12.07	16.89	59:59.99
		2	1.47	51.94	2183	29686	07:41.88	16.07	59:59.99
40	4	3	2.01	35.12	6526	26722	13:42.15	13.94	59:59.99
		4	1.56	46.66	3443	18792	06:45.07	3.94	59:59.99
		5	0.96	56.95	12319	26858	11:43.16	4.62	59:59.99
		1	1.21	50.29	277	29906	02:14.51	33.43	59:59.99
		2	1.31	43.33	219	24090	01:49.35	41.90	59:59.99
40	6	3	1.17	41.70	2142	19132	05:02.11	23.83	59:59.99
		4	1.23	36.49	20	17812	01:00.76	35.51	59:59.99
		5	1.21	52.67	2064	21534	05:27.13	28.00	59:59.99
		1	1.55	3.55	98	2435	17.82	0.01	30.82
		2	1.47	0.51	57	2199	11.95	0.00	23.31
60	2	3	1.42	5.20	78	3721	19.41	0.01	37.09
		4	1.48	4.60	257	4112	34.20	0.01	50.39
		5	1.45	0.57	10	1310	05.56	0.00	13.56

Table 2: Naive κ-FSP results (continued).

$\lvert\mathcal{V}\rvert$	κ	seed	build	root	nodes	cuts	cut time	gap	solve
		1	2.75	63.96	4615	29066	17:56.51	18.90	59:59.99
		2	2.85	66.61	15	5934	39.09	0.00	01:49.04
60	4	3	2.89	60.35	640	44674	07:11.69	35.21	59:59.99
		4	2.80	60.95	2	15763	02:15.70	0.00	22:34.62
		5	2.94	65.64	3983	28325	15:23.02	21.02	59:59.99
		1	4.20	34.51	20	17461	02:45.69	33.38	59:59.99
		2	4.27	55.77	2	12123	48.48	55.45	59:59.99
60	6	3	4.22	60.74	1	29361	01:44.70	60.74	59:59.99
		4	4.17	52.85	2	21744	01:22.24	52.48	59:59.99
		5	4.24	63.80	10	15876	01:35.67	63.54	59:59.99
		1	6.53	2.32	3	5672	55.45	0.00	01:48.99
		2	7.09	7.11	1567	18023	11:47.70	0.01	16:14.55
80	2	3	6.84	2.65	190	7213	02:17.77	0.0	03:14.40
		4	6.96	5.32	1680	5933	09:15.50	0.01	11:43.56
		5	6.75	84.44	175	5017	01:56.09	0.01	02:47.71
		1	13.26	0.00	0	14123	03:45.72	0.00	27:11.85
		2	13.63	66.1	668	29861	18:43.10	35.47	59:59.99
80	4	3	14.28	64.76	961	26757	20:10.76	35.00	59:59.99
		4	19.03	0.00	0	10357	01:30.74	0.00	08:42.35
		5	8.14	65.35	4	17854	02:33.30	0.00	18:08.11
		1	12.64	62.68	0	19815	01:56.94	62.68	59:59.99
		2	12.53	63.80	0	17955	02:32.88	63.80	59:59.99
80	6	3	20.80	62.76	0	23135	03:46.96	62.76	59:59.99
		4	18.86	68.30	0	21510	03:13.00	68.30	59:59.99
		5	18.62	59.15	2	17657	02:56.45	59.12	59:59.99
		1	14.80	0.00	0	22351	19:06.89	0.00	59:59.99
		2	14.37	84.25	2011	8513	18:00.09	0.01	21:29.06
100	2	3	9.42	51.11	69	7829	01:49.30	0.01	02:55.51
		4	9.26	81.46	7895	13902	21:33.00	1.04	59:59.99
		5	10.15	60.34	2435	11480	15:53.70	0.01	19:18.83
		1	18.64	66.40	3	28555	07:10.36	66.10	59:59.99
		2	17.93	70.43	20	29250	09:25.83	70.23	59:59.99
100	4	3	29.94	67.35	0	30382	09:19.46	67.35	59:99.99
		4	29.38	66.08	90	19801	19:51.46	65.99	59:59.99
		5	29.56	65.01	0	25941	21:15.90	65.01	59:59.99
		1	46.05	73.41	0	17980	04:16.96	73.41	59:59.99
		2	45.19	65.85	0	16369	03:40.89	65.85	59:59.99
100	6	3	43.62	69.68	0	19443	04:19.91	69.68	59:59.99
		4	47.12	63.99	0	23761	04:48.09	63.99	59:59.99
		5	28.10	64.61	0	25167	03:35.13	64.61	59:59.99

Table 3: Reduced κ-FSP results.

$\lvert\mathcal{V}\rvert$	κ	seed	build	root	nodes	cuts	cut time	gap	solve	alg	flow
		1	00.06	0.00	0	27	00.03	0.00	00.08	0.06	0.02
		2	00.06	0.00	0	34	00.05	0.00	00.14	0.08	0.03
20	2	3	00.06	0.00	0	54	00.04	0.00	00.09	0.10	0.04
		4	00.06	0.00	0	50	00.05	0.00	00.11	0.08	0.03
		5	00.06	0.72	1	39	00.07	0.00	00.18	0.09	0.03
		1	00.06	0.00	0	37	00.11	0.00	00.41	0.17	0.04
		2	00.06	0.00	0	14	00.05	0.00	00.17	0.02	0.05
20	4	3	00.06	1.82	9	37	00.15	0.00	01.01	0.22	0.09
		4	00.06	0.00	0	15	00.07	0.00	00.49	0.22	0.07
		5	00.13	5.27	53	39	00.42	0.00	01.69	0.03	0.05
		1	00.06	0.00	0	11	00.12	0.00	00.78	0.03	0.10
		2	00.07	2.73	5	5	00.12	0.00	00.86	0.01	0.10
20	6	3	00.06	1.76	5	5	00.11	0.00	00.75	0.01	0.17
		4	00.06	1.39	2	5	00.10	0.00	00.69	0.01	0.18
		5	00.06	3.59	5	31	00.12	0.00	00.71	0.01	0.11

Table 3: Reduced κ-FSP results (continued).

$\lvert\mathcal{V}\rvert$	κ	seed	build	root	nodes	cuts	cut time	gap	solve	alg	flow
		1	00.37	0.00	0	266	00.59	0.00	01.28	1.73	0.22
		2	00.31	0.00	0	108	00.21	0.00	00.43	1.88	0.22
40	2	3	00.37	0.00	0	118	00.34	0.00	00.79	3.20	0.50
		4	00.30	0.00	0	79	00.25	0.00	00.69	1.52	0.39
		5	00.30	0.00	0	132	00.35	0.00	00.79	2.00	0.25
		1	00.39	15.89	1267	758	23.31	0.00	50.28	1.07	0.38
		2	00.32	7.28	293	360	06.88	0.00	21.65	02:18.86	0.45
40	4	3	00.41	7.34	384	112	05.71	0.00	14.12	08.56	0.72
		4	00.31	7.83	2445	374	29.49	0.01	51.14	31.25	0.66
		5	00.32	9.81	5376	567	01:05.35	0.01	01:49.86	6.72	0.68
		1	00.43	10.62	322	259	06.40	0.00	19.32	34.23	0.02
		2	00.33	8.66	630	131	09.64	0.00	23.62	9.00	0.02
40	6	3	00.41	7.67	388	111	06.45	0.00	18.11	3.56	0.04
		4	00.33	9.05	153	185	04.83	0.00	18.87	2.45	0.04
		5	00.33	11.99	321	141	06.28	0.00	17.25	0.03	0.04
		1	01.15	5.84	1	279	01.33	0.00	03.51	7.47	0.04
		2	01.17	0.17	6	404	01.95	0.00	03.90	8.02	0.12
60	2	3	01.08	0.01	0	653	02.87	0.01	05.15	9.07	0.09
		4	01.16	0.72	18	561	03.36	0.01	07.57	9.50	0.11
		5	01.18	0.00	0	103	00.68	0.00	01.56	9.28	0.06
		1	01.18	9.10	7278	2109	03:41.88	0.01	10:30.60	02:44.65	0.15
		2	01.28	12.09	4595	2728	02:24.17	0.01	06:07.97	1:25.97	0.15
60	4	3	01.27	15.31	10031	8817	05:53.77	0.01	26:14.31	3:46.30	0.14
		4	01.38	18.26	14158	9889	11:02.35	3.90	59:59.99	27:35.28	0.31
		5	01.20	12.03	28609	2069	12:25.65	0.01	22:16.73	2:30.70	0.47
		1	01.67	10.03	780	1322	38.10	0.00	02:23.35	6:37.45	0.16
		2	01.28	11.26	249	345	14.61	0.00	52.30	12.11	0.36
60	6	3	01.47	17.20	3581	3732	02:28.77	0.00	14:59.62	7.38	0.31
		4	01.44	16.66	15589	5641	07:22.76	0.01	26:11.81	4:25.19	0.49
		5	01.53	11.99	588	495	29.28	0.00	01:47.08	3.10	0.59
		1	03.18	2.74	4	993	10.24	0.01	16.93	28.14	0.51
		2	02.99	2.72	63	3162	27.43	0.01	40.75	2:16.47	0.98
80	2	3	03.10	7.12	4	536	03.96	0.00	07.22	37.99	0.60
		4	03.10	42.42	20	1336	11.64	0.00	23.27	34.96	0.86
		5	02.98	0.00	0	418	03.33	0.00	06.70	32.98	0.68
		1	03.19	22.79	4597	16044	12:16.16	14.82	59:59.99	25:19.96	1.00
		2	03.32	18.28	560	3187	06:00.39	10.64	59:59.99	22:45.52	0.02
80	4	3	03.45	14.30	12436	30947	15:10.27	3.37	59:59.99	14:17.39	0.03
		4	02.31	19.35	7726	14908	08:53.25	9.58	59:59.99	—	0.04
		5	02.52	15.66	25500	3193	25:38.70	5.96	59:59.99	—	0.03
		1	03.96	19.74	2203	16514	08:47.92	6.80	59:59.99	—	0.03
		2	03.85	28.51	2444	17074	09:56.62	7.28	59:59.99	—	0.07
80	6	3	04.94	18.72	2472	9185	09:52.44	8.14	59:59.99	—	0.06
		4	04.72	19.69	2293	9162	06:44.81	5.66	59:59.99	—	0.09
		5	02.05	14.27	6595	3190	03:50.63	0.00	25:50.09	—	0.06
		1	04.22	0.42	52	2627	26.46	0.00	39.70	1:05.91	0.09
		2	03.99	2.55	39	1635	16.95	0.00	28.76	1:17.60	0.04
100	2	3	04.03	0.00	0	815	04.64	0.00	06.98	1:26.08	0.16
		4	04.11	2.31	24	544	04.15	0.00	09.77	1:13.05	0.28
		5	04.03	0.00	0	913	06.82	0.00	12.37	—	0.10
		1	04.11	22.69	4994	13485	11:26.42	17.02	59:59.99	—	0.64
		2	04.28	17.11	6416	14399	13:20.69	11.67	59:59.99	—	0.72
100	4	3	04.22	20.58	4757	15988	11:04.10	9.79	59:59.99	—	0.52
		4	04.28	17.78	17900	7152	24:12.56	9.28	59:59.99	—	0.74
		5	04.33	17.33	6195	17836	12:04.82	8.44	59:59.99	—	0.55
		1	04.48	29.49	4536	10450	09:42.24	18.09	59:59.99	—	1.52
		2	04.36	25.45	9548	10217	14:58.94	14.88	59:59.99	—	1.33
100	6	3	04.42	15.68	2268	11225	09:35.99	9.02	59:59.99	—	1.25
		4	09.10	17.15	8188	8913	25:44.44	14.17	59:59.99	—	1.22
		5	08.14	22.41	2600	8151	11:19.13	8.60	59:59.99	—	1.17

Table 4: Comparison of results for naive and reduced κ-FSP.

instance		**naive**					**reduced**				
$\lvert\mathcal{V}\rvert$	κ	build	cuts	cut time	opt.	solve	build	cuts	cut time	opt.	solve
	2	0.07	187	00.19	5	00.60	0.06	40	00.05	5	00.12
20	4	0.10	718	02.96	5	06.89	0.07	28	00.16	5	00.75
	6	0.16	656	03.19	5	08.28	0.06	11	00.11	5	00.76
	2	0.39	912	01.87	5	04.27	0.33	140	00.35	5	00.80
40	4	1.35	–	–	0	–	0.35	434	26.15	5	49.41
	6	1.23	–	–	0	–	0.37	165	06.72	5	19.43
	2	1.47	2755	17.79	5	31.03	1.15	400	02.04	5	04.34
60	4	2.85	10848	01:27.39	2	12:11.83	1.26	3930	06:06.37	4	16:17.40
	6	4.22	–	–	0	–	1.48	2307	02:14.70	5	09:14.83
	2	6.83	8371	05:14.50	5	07:09.84	3.07	1289	11.32	5	18.97
80	4	13.67	14111	02:36.59	3	18:00.77	2.96	–	–	0	–
	6	16.69	–	–	0	–	3.90	3190	03:50.63	1	25:50.09
	2	11.60	9274	11:54.36	3	14:34.47	4.08	1306	11.80	5	19.52
100	4	25.09	–	–	0	–	4.24	–	–	0	–
	6	42.02	–	–	0	–	6.10	–	–	0	–

6 Discussion and Conclusions

In this paper, two integer linear programming formulations of the κ-FSP have been explored. This problem is distinguished from existing literature by its focus on survivability, an essential feature in designing robust networks. Section 4 covered the equivalence of the formulations for cases of $\kappa = |\mathcal{R}|$, allowing for direct comparisons between them. Both the naive and reduced κ-FSP formulations can be solved using a similar branch-and-cut algorithm. However, the results from the previous section indicate that the reduced κ-FSP formulation outperforms the naive κ-FSP formulation consistently.

The largest instance where an optimal solution was found using the reduced κ-FSP formulation hade $|\mathcal{V}| = 80$ and $\kappa = 6$. While only one of the five runs found an optimal solution in this instance, the remaining runs found a feasible solution with a MIP gap less than 10%, suggesting that the solver may have reached optimality soon after a one-hour time limit. The largest instance solved using the naive κ-FSP formulation was $|\mathcal{V}| = 100$ and $\kappa = 2$, however, the performance of this formulation is inconsistent. Notably, no optimal solutions were found for $|\mathcal{V}| = 40$, $\kappa = 4$. The solver struggles more with the naive formulation as the number of roots in the problem increases, rather than the amount of vertices. This is symptomatic of its design: defining decision variables x_{ij}^r for each $(i, j, r) \in \mathcal{A} \times \mathcal{R}$ increases the problem size approximately by a factor of $|\mathcal{R}|$ compared to the reduced formulation.

When both formulations yield optimal solutions, the reduced κ-FSP significantly outperforms the naive formulation. For example, when $|\mathcal{V}| = 100$ and $\kappa = 2$, the solver found an optimal solution using the reduced formulation with an average solve time of under 20 seconds across all five runs, while the naive formulation had an average solve time of almost 15 minutes for the same instance.

Note that the performance of both formulation often varies and there are instances when the naive formulation produces an optimal solution and the reduced formulation does not, despite the same seed being used. Further experiments would provide greater insight to these differences.

The solution provided by the reduced κ-FSP formulation does not distinguish between directed trees within the forest. To address this, the solution was processed through the TREE_DECOMPOSITION algorithm and the flow model. The flow model performed consistently well across all problem sizes, with a maximum solve time of under two seconds. The TREE_DECOMPOSITION algorithm was effective for smaller problems, however as the number of arcs increases, the solve time increased exponentially and in some cases it failed to provide a solution due to memory limitations. While the algorithm offers a structured approach for decomposing solutions, the flow model is more practical due to its efficiency.

A recurring feature in the results for both formulations is the high MIP gap at root nodes, particularly for instances when $|\mathcal{R}|$ is larger. This could be attributed to the absence of a primal heuristic for the κ-FSP, leading to weaker initial bounds in the branch-and-cut algorithm, which could increase the overall solve time. Future research could explore the development of a primal heuristic. Given that a heuristic exists for the closely-related MSAP, insights from that problem could guide heuristic design for the κ-FSP.

While this paper proves equivalence of the naive and reduced formulations for $\kappa = |\mathcal{R}|$, the reduced formulation could produce a solution that is valid even when $\kappa < |\mathcal{R}|$. Proposition 1 does not rely on the restriction that $\kappa = |\mathcal{R}|$. A future direction of research could involve formally proving the remaining result.

The reduced formulation significantly reduces the size of the problem but removes the explicit assignment of paths in a solution. Despite this, it has been demonstrated that this essential practical information can be efficiently extracted through auxiliary methods. Given its ability to solve larger instances more quickly, the reduced formulation presents a clear advantage, making it a more practical choice for solving the κ-FSP in real-world applications.

References

[1] Cristina G. Fernandes and Carla N. Lintzmayer. "How heavy independent sets help to find arborescences with many leaves in DAGs". In: *Journal of Computer and System Sciences* 135 (2023), pp. 158–174. ISSN: 0022-0000. DOI: https://doi.org/10.1016/j.jcss.2023.02.006. URL: https://www.sciencedirect.com/science/article/pii/S0022000023000260.

[2] Vinicius Morais, Bernard Gendron, and Geraldo Robson Mateus. "The p-arborescence star problem: Formulations and exact solution approaches". In: *Computers Operations Research* 102 (2019), pp. 91–101. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2018.10.004. URL: https://www.sciencedirect.com/science/article/pii/S0305054818302570.

[3] Gourab Ray and Arnab Sen. *Minimal spanning arborescence*. 2024. arXiv: 2401.13238 [math.PR]. URL: https://arxiv.org/abs/2401.13238.

[4] Jack Edmonds. "Optimum branchings". In: (1967). DOI: 10.6028/jres.071b.032.

[5] L. Simonetti, Y. Frota, and C.C. de Souza. "The ring-star problem: A new integer programming formulation and a branch-and-cut algorithm". In: *Discrete Applied Mathematics* 159.16 (2011). 8th Cologne/Twente Workshop on Graphs and Combinatorial Optimization (CTW 2009), pp. 1901–1914. ISSN: 0166-218X. DOI: https://doi.org/10.1016/j.dam.2011.01.015. URL: https://www.sciencedirect.com/science/article/pii/S0166218X11000278.

[6] Martine Labbé et al. "The Ring Star Problem: Polyhedral analysis and exact algorithm". In: *Networks* 43.3 (2004), pp. 177–189. DOI: https://doi.org/10.1002/net.10114. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.10114. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.10114.

[7] Robin J Wilson. *Introduction to graph theory*. USA: John Wiley & Sons, Inc., 1986. ISBN: 0470206160.

[8] Thomas H. Cormen et al. *Introduction to Algorithms, Third Edition*. 3rd. The MIT Press, 2009. ISBN: 0262033844.

[9] L. R. Ford and D. R. Fulkerson. "Maximal Flow Through a Network". In: *Canadian Journal of Mathematics* 8 (1956), pp. 399–404. DOI: 10.4153/CJM-1956-045-5.