Strong ILP Formulations for the p-Regions Problem

Daniel Faber
University of Bonn, Germany

Jan-Henrik Haunert
University of Bonn, Germany

Petra Mutzel
University of Bonn, Germany
Lamarr Institute, Bonn, Germany

Abstract

Regionalization is a fundamental task in spatial analysis that seeks to partition a larger area - such as a country - into smaller regions that are homogeneous with respect to a given attribute. A popular model for regionalization is the p-regions problem, in which regions are formed by grouping the areas of an input planar subdivision. Given the subdivision's adjacency graph G and pairwise dissimilarities between vertices, the goal is to partition G into a fixed number p of connected subgraphs, such as to minimize the sum of dissimilarities over all vertex pairs in the same subgraph. The problem is NP-hard and even small instances are difficult to solve to provable optimality.

In this paper, we present the new ILP model `ER-S` for the p-regions problem, exploiting a connection between the p-regions objective and the k-partitioning problem. Furthermore, we strengthen the known ILP model `Tree` with a new type of subtour elimination inequality specific to the p-regions problem. Combining `ER-S` and the strengthened version of `Tree` yields the model `ER-S-Tree`, which dominates the state-of-the-art models in polyhedral strength. This theoretical advantage is reflected in its superior performance in our experimental evaluation. In particular, the new models `ER-S` and `ER-S-Tree` enable the solution of problem instances for major European countries that were previously intractable.

2012 ACM Subject Classification Theory of computation → Discrete optimization

Keywords and phrases p-regions problem, connected graph partitioning, area aggregation, integer linear programming, branch-and-cut

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.13

Related Version *Full Version*: `https://arxiv.org/abs/2607.04886`

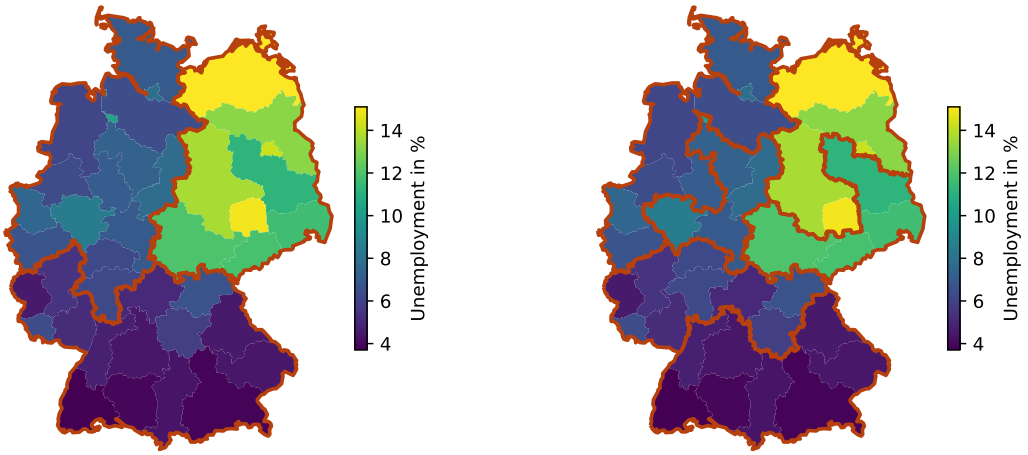
Supplementary Material *Software*: `https://zenodo.org/records/21275790` [13]

Funding This research was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant FOR-5361 – 459420781.

Acknowledgements We want to thank Michael Kaibel for his helpful remarks.

1 Introduction

The aggregation of spatial data is a core topic in geoinformation science, and has given rise to many computationally challenging problems. A common restriction in these aggregation problems is regional connectivity: Given a set of geographic areas, the goal is to group similar (with respect to a given attribute) areas into a smaller number of regions, where each region needs to be contiguous. There are numerous problems that fall into this class of connected clustering problems, such as area aggregation in map generalization [22] and political districting [23]. Almost all of these problems are NP-hard, and solving them to



(a) $k = 3$. **(b)** $k = 6$.

Figure 1 Regionalization solutions for the NUTS-2 subdivison of Germany for $k \in \{3, 6\}$.

optimality has proven to be a difficult task. Integer Linear Programming (ILP) techniques have shown great success for many difficult combinatorial optimization problems, and thus the majority of exact algorithms in the area of connected clustering problems are based on ILP formulations.

One instance of an NP-hard connected clustering problem is the p-regions problem, introduced by Duque et al. [8]. They consider the problem of clustering a set of given areas into a predefined number of p spatially contiguous regions, while minimizing the total heterogeneity of all regions. The authors define the heterogeneity of a region by the sum of dissimilarities between all pairs of vertices within a region, where the dissimilarity function $d(u, v)$ for all pairs of areas u, v is given as part of the problem input. Figure 1 shows solutions to the p-regions problem for unemployment data of Germany, with $k \in \{3, 6\}$.

In their work, the authors frame the p-regions problem as a graph problem and propose the three ILP models `Tree`, `Order` and `Flow` [8]. Their computational experiments show that even small instances (# areas ≈ 25) are already challenging and could not be solved within the time limit of 3 hours.

The p-regions problem is closely related to both the k-partitioning problem and connected clustering problems. In the k-partitioning problem, the goal is to partition the vertices of an edge-weighted graph into k subsets such that the total weight of edges whose endpoints lie in the same part is minimized. It is equivalent to the max-k-cut problem, where the goal is to maximize the weight of edges running between partitions. Notably, Chopra et al. [6, 7], Ales et al. [2] and Fairbrother et al. [14] studied the problem from a polyhedral perspective and identified various facet-defining inequalities. They consider formulations with only edge variables, edge and representative variables, and edge and vertex assignment variables.

Recently, there have been a number of ILP approaches for the connected max-k-cut problem, which is defined as the max-k-cut problem with the additional constraint that each partition must be connected [17, 16]. Notably, Healy et al. [16] proposed an ILP model based on edge cut constraints. In contrast to the p-regions problem, where the costs are defined over every pair of vertices, the objective is defined over the cost of cut edges. In the p-regions problem, assigning two non-adjacent vertices to different parts is associated with a cost, whereas the connected max-k-cut problem does not penalize such assignments. Therefore the p-regions problem is more general.

Another closely related problem is the *graph-connected clique-partitioning problem (GCCP)*, introduced in [4]. As in the p-regions problem, the cost function is defined over each vertex pair and the objective is to minimize the sum of costs over all vertex pairs in the same partition. The difference between the two problems is that there is no constraint on the number of partitions for GCCP, while for the p-regions problem the number of regions is fixed. In [4], Benati et al. introduce three ILP formulations, based on a classic flow formulation, a formulation based on Miller-Tucker-Zemlin constraints and a cut-based formulation. In a subsequent paper, they propose a branch-and-price approach for this problem [3].

Our Contribution

In this work, we present and theoretically analyze new ILP formulations for the p-regions problem. Based on these formulations, we develop a branch-and-cut algorithm and evaluate its performance experimentally in a computational study, demonstrating significant improvements over the state of the art. In particular:

- We propose a new ILP formulation `ER-S` for the p-regions problem exploiting a connection to the k-partitioning problem. Specifically, we combine the edge-representative model for the k-partitioning problem [2] and the vertex-separator constraints for modeling graph connectivity [22].
- We present constraints to strengthen `ER-S` and `Tree` [8] as well as a combination of the strengthened models into a new hybrid model `ER-S-Tree`.
- We conduct a polyhedral analysis of our new models and known formulations from the literature [8], `Tree` and `Flow`, showing that our new hybrid model `ER-S-Tree` strictly dominates.
- We provide implementations of our branch-and-cut algorithms based on the formulations `ER-S-Tree`, `ER-S`, `Tree`, and `Flow`, which are publicly available [13].
- We experimentally evaluate the performance of our new algorithms on a benchmark set of real-world instances as well as generated grid instances for multiple numbers of regions. `ER-S` and `ER-S-Tree` display superior performance across the benchmark set, supporting our theoretical findings. In particular, neither `Flow` nor `Tree` was able to solve the instance `Romania` for any tested value for k, while `ER-S-Tree` manages to solve it for all but one of the eight tested values for k.

The rest of this paper is organized as follows: In section 2, we introduce the notation used in this paper and formally define the p-regions problem. In section 3, we revisit state-of-the-art ILP formulations for the p-regions problem and related problems that our formulations build upon. In section 4, we present a strengthened version of the model `Tree` proposed in [8]. Section 5, introduces our new models `ER-S` and `ER-S-Tree`. The relaxation strength of the models is analyzed in section 6. In section 7, we describe the separation routines of our branch-and-cut algorithms. The results of our experimental evaluation are discussed in section 8. Finally, 9 concludes the paper.

2 Preliminaries

We use the common notation $\binom{V}{i}$ to describe the set of all subsets of V that have cardinality i. We also use the notations $[b]$ and $[a, b]$ for $b \in \mathbb{N}_0$ to denote the set $\{1, \dots, b\}$ and $\{a, \dots, b\}$ of natural numbers.

In this paper, we mainly consider simple undirected graphs $G = (V(G), E(G))$ with dissimilarities $d\colon \binom{V}{2} \to \mathbb{R}$ defined between any pair of vertices. Following standard notation, we define $n = |V(G)|, m = |E(G)|$. We consider the set of vertices to be numbered, i.e. $V = [n]$. We will use V and E instead of $V(G)$ and $E(G)$ if G is clear from the context.

For a vertex set $V' \subseteq V$, $G[V']$ denotes the subgraph induced by V', with vertex set V' and edge set $E(V') = \{e \in E \mid e \subseteq V'\}$. For an undirected graph $G = (V, E)$, we define the neighborhood $N(v) = \{w \in V(G) \mid \{v, w\} \in E(G)\}$ For a directed graph $D = (V, A)$, we define $N^-(v) = \{u \in V(D) \mid (u, v) \in A(D)\}$ and $N^+(v) = \{w \in V(D) \mid (v, w) \in A(D)\}$.

Since some algorithms and formulations are defined only for directed graphs, we associate with a set of undirected edges E the set of directed edges $A(E) \subseteq V \times V$ containing both (u, v) and (v, u) for every undirected edge $\{u, v\} \in E$.

We now formally define the p-regions problem. We remark that in the following, we denote the number of regions requested as output by k (as opposed to p), as this is standard convention in the graph partitioning literature [2, 16, 6, 14].

▶ **Definition 1** (p-regions problem). *Given a connected graph $G = (V, E)$, pairwise vertex dissimilarities $d\colon \binom{V}{2} \to \mathbb{R}_{\geq 0}$ and an integer $k \in [n]$, the p-regions problem is to find a k-partitioning $\mathcal{P} = (V_1, ..., V_k)$ of V such that*

1. *$V = \bigcup_{i \in [k]} V_i$ (each vertex is covered)*
2. *$V_i \cap V_j = \emptyset$ for $i \neq j$ (the partitions are disjoint)*
3. *$G[V_i]$ for $i \in [k]$ is connected*
4. *$V_i \neq \emptyset$ for all $i \in [k]$*

minimizing the total heterogeneity $H(\mathcal{P}) = \sum_{i \in [k]} \sum_{u,v \in V_i} d(u, v)$.

3 State-of-the-art ILP formulations

3.1 ILP formulations for the p-regions problem

In this section we review the ILP models for the p-regions problem proposed by Duque et al. [8]. In this work, the authors proposed three ILP models, namely `Flow`, `Tree` and `Order`. We will discuss the first two models, as `Order` has been shown to not be competitive [8].

3.1.1 Formulation `Flow`

Formulation `Flow` [8] (see Model 1) contains three sets of binary and one set of continuous variables. The first set consists of binary variables $z_{v,i}$ for each vertex $v \in V$ and region $i \in [k]$, where $z_{v,i} = 1$ iff $v \in V_i$. Additionally, the model has binary variables $x_{u,v}$ for each vertex pair $u, v \in V$ with $u \neq v$, defined by $x_{u,v} = 1$ exactly if u and v are in the same region (denoted as *pairing variables*). To enforce connectivity within a region, the model uses a standard multi-commodity flow formulation, using flow variables $f^i_{u,v}$ for $(u, v) \in A(E), i \in [k]$ and binary root variables $r_{u,i}$ for $u \in V, i \in [k]$.

The objective minimizes the total sum of dissimilarities between vertices within the same region in the partitioning. Constraints (1) ensure that $x_{u,v} = 1$ if there exists an $i \in [k]$ such that $z_{u,i} = z_{v,i} = 1$. Finally, constraints (2) enforce that each vertex is assigned to exactly one region. Constraints (3) – (7) model regional connectivity via a flow network for each region. Constraints (3) – (4) ensure that for each partition, exactly one vertex is assigned as the root vertex. Constraints (5) – (6) ensure that flow of type $i \in [k]$ can only traverse an edge if both incident vertices are assigned to i. Here, M denotes a sufficiently large constant. The authors use $M = n - k + 1$, as a partition can have at most $n - k + 1$ vertices. Finally, the flow conservation constraints (7) enforce that each vertex assigned to partition i that is not the root must consume exactly one unit of flow. Constraint (8) has the purpose of breaking some symmetries arising from permuting the region indices.

$$
\begin{aligned}
\min \quad & \sum_{u<v} d(u,v) \cdot x_{u,v} && && \\
\text{s.t.} \quad & x_{u,v} \geq z_{u,i} + z_{v,i} - 1 && \forall u,v \in V, u \neq v, i \in [k] && (1)\\
& \sum_{i \in [k]} z_{v,i} = 1 && \forall v \in V && (2)\\
& \sum_{u \in V} r_{u,i} = 1 && \forall i \in [k] && (3)\\
& r_{u,i} \leq z_{u,i} && \forall u \in V, i \in [k] && (4)\\
& f^i_{u,v} \leq (M-1) \cdot z_{u,i} && \forall (u,v) \in A(E), i \in [k] && (5)\\
& f^i_{u,v} \leq (M-1) \cdot z_{v,i} && \forall (u,v) \in A(E), i \in [k] && (6)\\
& \sum_{u \in N(v)} f^i_{u,v} - \sum_{w \in N(v)} f^i_{v,w} \geq z_{v,i} - M \cdot r_{v,i} && \forall v \in V, i \in [k] && (7)\\
& z_{1,1} = 1 && && (8)\\
& x_{u,v} = x_{v,u} && \forall (u,v) \in V \times V, u \neq v && (9)\\
& z_{v,i}, r_{v,i} \in \{0,1\} && \forall v \in V, i \in [k] && (10)\\
& x_{u,v} \in \{0,1\} && \forall (u,v) \in V \times V, u \neq v && (11)\\
& f^i_{u,v} \in [0, M-1] && \forall (u,v) \in A(E), i \in [k] && (12)
\end{aligned}
$$

Model 1 Model `Flow` proposed in [8].

3.1.2 Formulation `Tree`

The tree formulation [8] (see Model 2) models each region as a directed arborescence (oriented towards the root) in the graph and is defined over the directed edge set $A(E)$. Like formulation `Flow`, the model contains pairing variables $x_{u,v}$. Additionally, it uses linking variables $y_{u,v}$ for all $(u,v) \in A(E)$, where $y_{u,v} = 1$ exactly if (u,v) is part of an arborescence. To enforce that the selected edges form k trees spanning all vertices, one can fix the number of selected edges to $n-k$ and ensure that the set of selected edges do not contain any cycle. The authors propose two variants of cycle-breaking constraints, a compact encoding and a formulation with up to exponentially many cycle breaking constraints. We will only consider the latter one, as the compact formulation was shown to be non-competitive.

The objective is equivalent to the one in `Flow`. Constraint (13) enforces that exactly $n-k$ edges are selected, the total number of edges contained in k edge-disjoint trees spanning all vertices. Constraints (14) enforce that each vertex can have at most one outgoing arc, as the arborescences are oriented towards the root. Constraints (15) are the so-called *transitivity constraints*: They ensure that for any triple of vertices u, v, w, if u, v and v, w are in the same region, then u and w are also in the same region, yielding a consistent pairing. Constraints (16) state that an edge can only be selected if the incident vertices are in the same region. Finally, the subtour elimination constraints (17) ensure that for any set $S \subseteq V$ the set of selected edges never form a cycle in $G[S]$ by imposing that $G[S]$ contains at most $|S| - 1$ selected edges.

$$\begin{aligned}
\min \quad & \sum_{u<v} d(u,v) \cdot x_{u,v} & & \\
\text{s.t.} \quad & \sum_{(u,v)\in A(E)} y_{u,v} = n - k & & (13)\\
& \sum_{v\in N(u)} y_{u,v} \leq 1 & \forall u \in V & (14)\\
& x_{u,w} \geq x_{u,v} + x_{v,w} - 1 & \forall u,v,w \in \binom{V}{3} & (15)\\
& y_{u,v} \leq x_{u,v} & \forall (u,v) \in A(E) & (16)\\
& \sum_{(u,v)\in A(S)} y_{u,v} \leq |S| - 1 & \forall S \subseteq V & (17)\\
& x_{u,v} = x_{v,u} & \forall (u,v) \in V \times V, u \neq v & (18)\\
& y_{u,v} \in \{0,1\} & \forall (u,v) \in A(E) & (19)\\
& x_{u,v} \in \{0,1\} & \forall (u,v) \in V \times V, u \neq v & (20)
\end{aligned}$$

Model 2 Model `Tree` proposed in [8].

3.2 ILP formulation for k-partitioning

The k-partitioning problem, as defined by Ales et al. [2], is as follows: Given an edge-weighted complete graph $H = (V(H), E(H))$ with weights $d\colon E(H) \to \mathbb{R}$, find a partitioning of the vertices into k parts minimizing the total weight of edges whose endpoints lie in the same part. Ales et al. [2] proposed two ILP formulations, the *node-cluster* formulation and the *edge-representative* formulation. As the node-cluster formulation suffers from a poor LP relaxation and symmetries, we present the edge-representative formulation `ER` (see Model 3).

$$\begin{aligned}
\min \quad & \sum_{u<v} d(u,v) \cdot x_{u,v} & & \\
\text{s.t.} \quad & x_{u,w} \geq x_{u,v} + x_{v,w} - 1 & \forall u,v,w \in \binom{V}{3} & (21)\\
& r_v \leq 1 - x_{u,v} & \forall u,v \in V, u < v & (22)\\
& r_v \geq 1 - \sum_{u<v} x_{u,v} & \forall v \in V & (23)\\
& \sum_{v\in V} r_v = k & & (24)\\
& x_{u,v} = x_{v,u} & \forall (u,v) \in V \times V, u \neq v & (25)\\
& r_v \in \{0,1\} & \forall v \in V & (26)\\
& x_{u,v} \in \{0,1\} & \forall (u,v) \in V \times V, u \neq v & (27)
\end{aligned}$$

Model 3 Model `ER` proposed in [2].

Like `Tree`, the formulation contains pairing variables $x_{u,v}$ and transitivity constraints (21). Additionally, the formulation contains binary variables r_v for each $v \in V$, which we denote as *representative variables*.

The underlying idea is that we designate exactly one vertex per part as a representative, so that we can constrain the number of parts by constraining the number of representatives. By convention, a part is represented by the vertex in the part having the lowest index. By constraints (22) and (23), it holds that $r_v = 1$ exactly if v is paired with no other vertex u with $u < v$ (in which case we call v a representative): (22) enforces that a vertex v cannot be a representative if it is paired with a vertex $u < v$, while (23) enforces that if no vertex with smaller index is paired with v, v must be a representative. Therefore, there must be exactly one representative per region, so the number of regions can be restricted by fixing $\sum_{v \in V} r_v = k$ in constraint (24).

3.3 ILP formulations for spatial contiguity

Oehrlein and Haunert [22] consider a similar districting problem, which also requires a partitioning of the adjacency graph G into connected regions. In contrast to our problem, they use an objective based on region centers instead of our pairwise objective and do not prescribe the number of output regions. Nevertheless, the connectivity constraints used in their model can still be applied to our setting. The constraints are based on the concept of vertex separators, defined as follows:

▶ **Definition 2** (u, v-Separator). *Given a graph $G = (V, E)$ and vertices $u, v \in V$, a u, v-Separator $S \subseteq V$ is a set of vertices such that $G[V \setminus S]$ does not contain a path between u and v.*

A graph is connected exactly if for any $u, v \in V$, every u, v-separator is non-empty. Therefore, connectivity can be expressed using the following constraints:

$$x_{u,v} \leq \sum_{w \in S} x_{u,w} \qquad \forall u, v \in V, u \neq v, S \in \mathcal{S}_{u,v} \tag{28}$$

Here, $\mathcal{S}_{u,v}$ is the set of all u, v-separators, while variables $x_{u,v} = 1$ exactly if vertices u and v are paired together.

4 `Tree+`: Strengthening Formulation `Tree`

Before introducing our new algorithm `ER-S` in the next section, we propose two modifications of `Tree` presented in [8, 10].

The first modification is strengthening constraints (16) by substituting it with

$$y_{u,v} + y_{v,u} \leq x_{u,v} \qquad \forall \{u, v\} \in E \tag{29}$$

The idea is that the directed edges (u, v) and (v, u) cannot be part of the same arborescence. The resulting constraints are at least as strong as (16) as $y_{u,v} \leq y_{u,v} + y_{v,u}$.

The second modification is a new type of valid inequality based on the cycle-breaking constraints (17):

$$2 \cdot \sum_{(u,v) \in A(T)} y_{u,v} - \sum_{\{u,v\} \in T} x_{u,v} \leq |S| - 2 \qquad \forall S \subseteq V, T \subseteq E(S) : |T| \geq |S| \tag{30}$$

▶ **Theorem 3.** *Every valid integer solution (x^*, y^*) of formulation `Tree` satisfies constraints (30).*

Proof. Let $S \subseteq V$ and $T \subseteq E(S)$ with $|T| \geq |S|$ and let $F = \{a \in A(T) \mid y^*_a = 1\}$. As F induces a forest in G and by extension $G[S]$, it holds that $|F| \leq |S| - 1$. We distinguish between two cases:

Case 1: $|F| \leq |S| - 2$: By constraints (16), we have

$$\sum_{(u,v)\in A(T)} y^*_{u,v} - \sum_{\{u,v\}\in T} x^*_{u,v} \leq 0$$

from which constraint (30) follows.

Case 2: $|F| = |S| - 1$: Because, by construction, the edges in F cannot contain a cycle, they must form an arborescence spanning all vertices in S. Therefore by constraints (16) and the transitivity constraints (15), all vertices in S are paired together, which implies that $x^*_{u,v} = 1$ for all $\{u, v\} \in T$. As $|T| \geq |S|$, it must hold that $\sum_{\{u,v\}\in T} x^*_{u,v} = |T| \geq |S|$ which completes the proof, as it must hold that $2 \cdot \sum_{(u,v)\in A(T)} y^*_{u,v} \leq 2(|S| - 1)$. ◀

Furthermore, we can show that for sets S with $|E(S)| = |S|$ (e.g. chordless cycles), constraints (30) dominate constraints (17).

▶ **Observation 4.** *For vertex sets S with $|E(S)| = |S|$, constraints* (30) *imply constraint* (17).

For the proof, see Section A.3 of the appendix.

The resulting improved model `Tree+` is defined as Model 4.

$$\begin{aligned} &\text{min} && \sum_{u<v} d(u,v) \cdot x_{u,v} \\ &\text{s.t.} && (13) - (15), (17) - (20), (29), (30) \end{aligned}$$

Model 4 Model `Tree+`.

5 New ILP formulations `ER-S` and `ER-S-Tree`

5.1 `ER-S-Base`

One can observe that the p-regions problem essentially is a k-partitioning problem under the constraint that every region must be connected. It is important to note that, in this setting, the complete graph $H = (V, E(H))$ defining the objective and the adjacency graph $G = (V, E(G))$ defining the connectivity are independent from each other. Given the adjacency graph G and dissimilarities $d : \binom{V}{2} \to \mathbb{R}$, define the dissimilarity graph to be the complete graph $H = (V, E(H))$ with edge weights defined by d. The p-regions problem can then be stated as finding an optimal k-partitioning in H, with the additional constraint that each partition V_i must induce a connected subgraph $G[V_i]$ in G.

Following this motivation, we combine ideas from the presented state-of-the art ILP formulations for k-partitioning with formulations for enforcing regional connectivity. We propose a new ILP model for the p-regions problem, which is derived by combining the edge-representative model for k-partitioning [2] and the vertex separator constraints for connectivity [22]. As in the edge-representatives model for k-partitioning, we define binary variables $x_{u,v}$ (called *pairing variables*) for each $u, v \in V$ with $x_{u,v} = 1$ iff u and v share a

partition and binary representatives variables r_v for $v \in V$, with $r_v = 1$ iff v is a representative. The model `ER-S-Base` (**e**dge-**r**epresentatives model with **s**eparator constraints) is defined as Model 5.

$$\begin{aligned} &\min && \sum_{u<v} d(u,v) \cdot x_{u,v} \\ &\text{s.t.} && (21) - (28) \end{aligned}$$

Model 5 Model `ER-S-Base`.

The model contains $\mathcal{O}(n^2)$ variables, $\mathcal{O}(n^3)$ constraints of type (21)-(27) and up to $\mathcal{O}(2^n n^2)$ constraints of type (28). The correctness of the model directly follows from the correctness of the edge-representative model for the k-partitioning problem and the correctness of the vertex separator constraints. This model forms the basis of the following models `ER-S` and `ER-S-Tree`. Notably, it does not contain any symmetries like `Flow`, so we do not need to consider any symmetry breaking constraints.

5.2 ER-S

The model `ER-S-Base` already correctly defines the feasible solutions and objective of the p-regions problem. To strengthen the LP relaxation of the model, we additionally include further valid constraints.

k-Forest Constraint. The k-forest constraint follows from the observation that any connected partition contains a spanning tree with exactly $|V_k| - 1$ edges over its vertices.

As the incident vertices of these trees are paired together, the following inequality is valid:

$$\sum_{\{u,v\} \in E} x_{u,v} \geq n - k \tag{31}$$

General Clique Constraints. General clique constraints were introduced in Chopra et al. [6] for the k-partitioning problem and were proven to be facet defining for the k-partitioning polytope. They are based on the idea that in a set Q of $|Q| = qk + p$ vertices (with $q \in [\lfloor \frac{n}{k} \rfloor], p \in [0, k-1]$), minimizing the number of vertex pairs in Q that are in the same region is achieved by spreading the vertices of Q into p clusters of size $q+1$ and $k-p$ clusters of size q. The resulting lower bound on the number of paired vertices yields the following valid constraints:

$$\sum_{u,v \in Q} x_{u,v} \geq \binom{q+1}{2} p + \binom{q}{2} (k-p) \qquad \forall Q \subseteq V, |Q| = qk + p \geq k+1 \tag{32}$$

The constraints corresponding to subsets Q with $|Q| = k+1$ are also simply known as clique inequalities [6].

Formulation `ER-S`. We define the model `ER-S` (see Model 6) to be the model `ER-S-Base` strengthened with the general clique constraints and the k-forest constraint. The model contains the same set of variables as `ER-S-Base`. It has one additional constraint (31) and $\mathcal{O}(2^n)$ additional constraints of type (32).

$$\begin{aligned} \min \quad & \sum_{u<v} d(u,v) \cdot x_{u,v} \\ \text{s.t.} \quad & (21) - (28), (31), (32) \end{aligned}$$

Model 6 Model `ER-S`.

5.3 `ER-S-Tree`

To further tighten the model, we consider a combination of the models `ER-S` and `Tree+`, denoted as `ER-S-Tree` (see Model 7). Constraints (33) are a strengthened version of con-

$$\begin{aligned} \min \quad & \sum_{u<v} d(u,v) \cdot x_{u,v} \\ \text{s.t.} \quad & (21) - (28), (31), (32) \\ & (13), (15), (17) - (20), (29), (30) \\ & \sum_{v \in N(u)} y_{u,v} = 1 - r_u \qquad \forall u \in V \qquad (33) \end{aligned}$$

Model 7 Model `ER-S-Tree`.

straints (14), which exploit the presence of root variables in formulation `ER-S`. We consider the arborescenes to be rooted in the root vertices defined by variables r in `ER-S`. A vertex has exactly one outgoing arc in the arborescence if it is not a root and otherwise zero. This has the additional benefit of eliminating some symmetries that arise from the ambiguous choice of trees for each partition.

The polyhedral analysis in the next section demonstrates that combining the constraints of `ER-S` and `Tree+` into `ER-S-Tree` yields a model that can be strictly stronger than both.

6 Theoretical Analysis

In this section, we conduct a theoretical analysis of the different ILP formulations, proving that `ER-S-Tree` has a strong relaxation. We compare the relaxation strength of the SOTA formulations `Tree` and `Flow` [8] with our new formulations `ER-S` and `ER-S-Tree`. We will use the following notation: $P(\square) \subseteq \mathbb{R}^{|\square|}$ describes the set of feasible solutions for the LP relaxation of $\square \in \{\texttt{ER-S}, \texttt{ER-S-Tree}, \texttt{Tree}, \texttt{Flow}\}$ (where $|\square|$ is the number of variables in $\square$). Furthermore, for a model $\square$ and a set Δ of variables contained in $\square$, $P_\Delta(\square)$ describes the projection of $P(\square)$ onto the space of variables Δ.

In the following, we will compare the polytopes of the models with respect to their projection onto the pairing variables $P_x(\square)$ for the following three reasons: (1) These variables are the only variables relevant for the objective, (2) they are included in every model, (3) they completely describe a partitioning without any arbitrary symmetries (e.g. region indices). We assume that M is chosen to be $n - k + 1$ for `Flow`, as every lower value for M leads to an incomplete model (i.e. the model excludes feasible solutions).

We first remark that the relaxation of `Flow` is very weak, which can be seen from the following theorem:

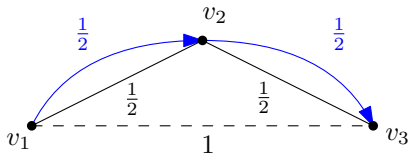


Figure 2 Example for $P(\texttt{Tree+}) \not\subset P(\texttt{ER-S})$, $k = 2$: the values of y are in blue, the values for x in black. Solid lines depict existing edges.

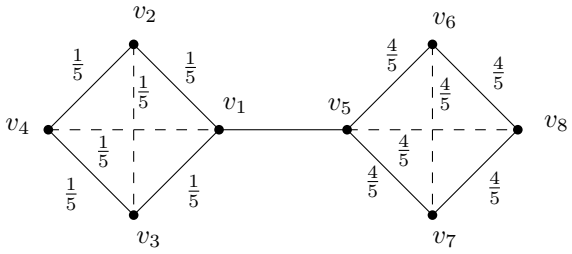


Figure 3 Example for $P(\texttt{ER-S}) \not\subset P(\texttt{Tree})$, $k = 4$. The figure depicts the non-zero values of x, existing edges are drawn as solid lines. The representative variables have values: $r_1, r_5 = 1, r_2 = \frac{4}{5}, r_3 = \frac{3}{5}, r_4 = \frac{2}{5}, r_6 = \frac{1}{5}, r_7, r_8 = 0$.

▶ **Theorem 5.** *For $k \geq 3$, $P_x(\texttt{ER-S}) \subseteq P_x(\texttt{Flow})$ and $P_x(\texttt{Tree}) \subseteq P_x(\texttt{Flow})$ and both inclusions can be strict. In particular, for any instance with $3 \leq k \leq n$, it holds that $P_x(\texttt{Flow}) = [0,1]^{(n^2-n)}$ and for $k < n$ the inclusion is always strict.*

Proof (Sketch). In a fractional solution, we can assign every vertex to be partially assigned to every region $i \neq 1$ by $z_{v,i} = \frac{1}{k-1}$. This results in the right-hand side of constraints (1) to be zero and the pairing variables to be unbounded. To show that these conclusion is strict for any $k < n$, we can observe that neither ER-S nor Tree contain the solution $x_{u,v} = 0$ for all $u, v \in V$. For the full proof, see Section A.1 of the appendix. ◀

▶ **Theorem 6.** *$P_x(\texttt{Tree+}) \subseteq P_x(\texttt{Tree})$ and this inclusion can be strict.*

Proof (Sketch). The inclusion trivially holds. To see that the inclusion can be strict, we can consider a triangle, and a fractional assignment of $\frac{2}{3}$ for every pairing and every arc in a directed cycle in the triangle. We observe that this satisfies constraints (17) but violates constraints (30). For the complete proof, see Sections A.2 of the appendix. ◀

▶ **Theorem 7.** *$P_x(\texttt{ER-S}) \not\subset P_x(\texttt{Tree})$ and $P_x(\texttt{Tree}) \not\subset P_x(\texttt{ER-S})$, i.e. the two models are incomparable under inclusion. The same holds for $P_x(\texttt{ER-S})$ and $P_x(\texttt{Tree+})$.*

Proof (Sketch). To show that $P_x(\texttt{Tree}) \not\subset P_x(\texttt{ER-S})$, we can consider the instance depicted in Figure 2 with a corresponding solution of Tree+ for $k = 2$. Notably, it violates the vertex separator constraints of ER-S.

For $P_x(\texttt{ER-S}) \not\subset P_x(\texttt{Tree})$, consider the example depicted in Figure 3 and the corresponding solution for ER-S for $k = 4$: To satisfy constraint (13), we would need to violate the subtour elimination constraint for $S = \{v_5, v_6, v_7, v_8\}$.

For the full proof, see Section A.4 of the appendix. ◀

▶ **Theorem 8.** *$P_x(\texttt{ER-S-Tree}) \subseteq P_x(\texttt{Tree+})$ and $P_x(\texttt{ER-S-Tree}) \subseteq P_x(\texttt{ER-S})$ and both inclusions can be strict.*

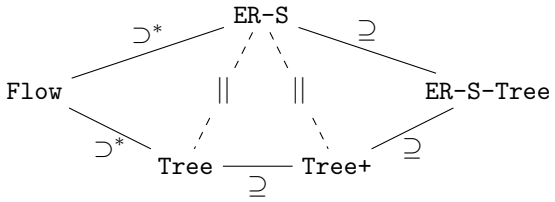


Figure 4 Inclusion hierarchy for the projected polytopes of the five analyzed models. The strict inclusion $\supset^*$ only holds for $3 \leq k \leq n-1$.

Proof. The inclusion $P_x(\texttt{ER-S-Tree}) \subseteq P_x(\texttt{ER-S})$ trivially follows from `ER-S-Tree` containing all constraints of `ER-S`. Similarly, $P_x(\texttt{ER-S-Tree}) \subseteq P_x(\texttt{Tree+})$ follows from `ER-S-Tree` containing all constraints of `Tree+`, with the exception of substituting constraints (14) by (33). However, (14) is implied by (33) as for the right-hand side of the inequalities it holds that $1 - r_v \leq 1$. The strict inclusions follow from Theorem 7. ◀

In summary, we have the inclusion hierarchy as depicted in Figure 4.

7 Separation algorithms

With the exception of `Flow`, all presented formulations contain a potentially exponential number of constraints. These require a branch-and-cut implementation, where we start by solving a relaxed version of the models omitting the constraint classes having exponential size and iteratively add them.

Subtour elimination constraints. Both `Tree` and `Tree+` contain the subtour elimination constraints (17). For separating integer solutions, we follow the approach by Duque et al. [10], which is to compute the set of strongly connected components induced by all arcs (u, v) with $y_{u,v} = 1$ in the current solution. Constraint (14) guarantees that there is a violated subtour elimination constraint exactly if there exists a strongly connected component of size at least 2.

For separating fractional solutions of the LP relaxation, we employ the classical algorithm by Magnanti and Wolsey [21] that is based on reducing the problem to a min-cut problem on an auxiliary graph.

To separate our new constraints (30), we use the following heuristic: using the above algorithm, we start with a set S minimizing $|S| - \sum_{(u,v)\in A(S)} y_{u,v}$. We then compute $T = \{\{u, v\} \in E(S) \mid 2(y_{u,v} + y_{v,u}) - x_{u,v} > 0\}$. If constraint (30) is violated for S and T, we add it to our model. We note that the algorithm by Magnanti and Wolsey cannot be directly applied to separate (30), as the associated costs $2(y_{u,v} + y_{v,u}) - x_{u,v}$ for an edge $\{u, v\}$ can be negative, prohibiting the reduction to the min-cut problem. We do not know if the separation problem is polynomial-time solvable or NP-hard.

Vertex separator constraints. The separation problem of constraints (28) for fractional solutions can be solved in polynomial time by transforming the problem of finding a minimum u, v-separator into a minimum u, v-cut problem. This can be done via a standard vertex-splitting graph transformation, see Oehrlein and Haunert [22]. For each pair of vertices, we run the algorithm of finding a minimum u, v-separator twice, once with costs $x_{u,a}$ for $a \in V \setminus \{u, v\}$ and once with costs $x_{v,a}$ for $a \in V \setminus \{u, v\}$. This results in $\mathcal{O}(n^2)$ calls to a flow algorithm on a graph of size $\mathcal{O}(n + m)$. Additionally, we use *nested cuts* [18], which is a technique to find multiple orthogonal cuts in one iteration.

General clique constraints. The separation of general clique constraints is NP-hard, as the independent set problem can be reduced to it. Therefore, we use a greedy heuristic incorporating local search steps to separate constraints (32). In each iteration, the algorithm greedily picks the best vertex to extend the current set S. It then checks if it can improve the current set by swapping out a vertex $b \in S$ with a vertex $a \in V \setminus S$. If the resulting set then violates the associated cut, it is added to the model. This is repeated until $|S| > limit$. The detailed pseudocode for the algorithm is given in Section B of the appendix (Algorithm 1).

8 Experimental evaluation

In our experimental evaluation, we are interested in answering the following questions:

- Q1: How does the performance of our algorithms `ER-S` and `ER-S-Tree` compare to the state-of-the-art algorithms `Flow` and `Tree` [8, 10] and to each other?
- Q2: How does the performance of each evaluated model depend on the predefined number of clusters k?

The algorithm was implemented using `C++20` and compiled using `gcc 13` with `-O3` optimization flag. We used `Gurobi 12.0` as the ILP solver and the `Boost Graph Library 1.89 (BGL)` for various graph algorithms. Further implementation details can be found in Section C of the appendix.

8.1 Benchmark Instances

Real-world instances. Our motivation for studying the p-regions problem was its application in map aggregation, consequently we are interested in real-world geostatistical data for our computational experiments. In that regard, we follow the experimental section of the work by Oehrlein et al. [22] and use data provided by the *European Statistical Office (Eurostat)* [11]. The data is acquired with respect to the NUTS (Nomenclature des unités territoriales statistiques) subdivision of Europe [12]. This subdivision contains three levels of hierarchy, with NUTS 1 being the coarsest and NUTS 3 the most fine-grained level of subdivision.

The geostatistical data we use in particular is the unemployment rate in Europe from 2010 [15], following Haunert et al. [22]. Table 5 gives an overview of the used instances. For all countries, we used the `NUTS-3` subdivison as the input subdivision, with the exception of `Germany`, where we used the `NUTS-2` subdivison as the size of the adjacency graph for `NUTS-3` was infeasible. For vertices $u, v \in V$, the dissimilarity was calculated as $d(u, v) = |\gamma(u) - \gamma(v)|$, where $\gamma(u)$ is the unemployment rate of area u.

Country	# Vertices	# Edges
`Finland`	19	41
`Bulgaria`	28	60
`United Kingdom`	36	74
`Greece`	38	71
`Germany`*	39	90
`Romania`	42	100
`Spain`	47	111

Figure 5 Overview of the instances used. The attribute used for each instance is the unemployment rate in each area. *For `Germany`, we used `NUTS-2` as the input subdivision.

Random instances. Following the works of Duque et al. on the p-regions problem [8, 10], we include another set of randomly generated instances simulating spatial auto-regressive processes on grid graphs. For a given integer *Size* and $\rho \in [0, 1]$, we generate a grid graph of size $Size \times Size$ with autocorrelation ρ and the rook contiguity criterion [19]. The details can be found in Section D of the appendix.

We generated one instance for every combination of $Size \in \{4, 5, 6, 7, 8, 9\}$ and $\rho \in \{0, 0.3, 0.6, 0.9\}$. Like for the real-world instances, the dissimilarity is calculated as $d(u, v) = |\gamma(u) - \gamma(v)|$, $\gamma(u)$ being the attribute of u.

8.2 Test Setup

We ran the experiments on an HPC cluster, with each node containing 2 × Intel Xeon "Sapphire Rapids" 48-core/96-thread 2.10GHz as CPUs and 1024GB of DDR5 4800MHz memory. The system is running the Linux distribution `AlmaLinux OS, Version 9.2 (Turquoise Kodkod)`. We set a time limit of 5 hours (18,000 s) for each instance.

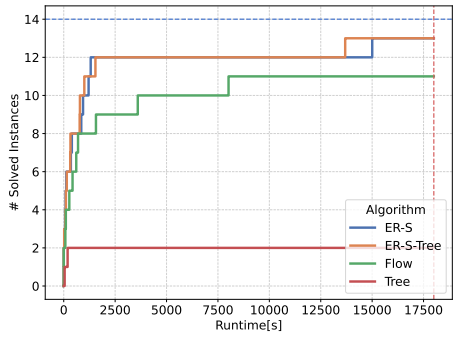
8.3 Results

Figure 6 shows the number of solved instances plotted against the runtime for `ER-S`, `ER-S-Tree`, `Flow` and `Tree`. The plots are grouped by different values for the number of predetermined regions k. Subfigures 6a - 6d show the results on the real-world instances, subfigures 6e - 6h on the random grid instances.

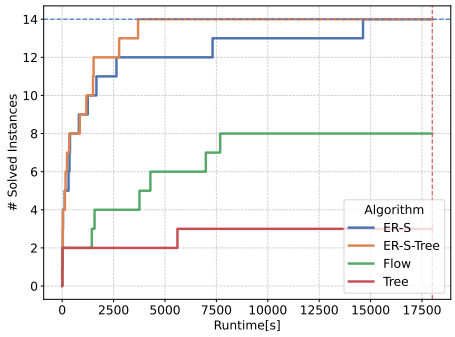
On the real-world instances (Subfigures 6a - 6d), we can see that our models `ER-S` and `ER-S-Tree` outperform both `Tree` and `Flow` for all values of k, especially for medium values ($k \in \{5, 6, 7, 8\}$). In accordance with the work by Duque et al. [8], we observe that `Flow` performs better for smaller values of k, while `Tree` performs better for larger values. For $k \in \{3, 4\}$, `Flow` is somewhat competitive with `ER-S` and `ER-S-Tree` (although still slower), whereas for all other values of k, it falls far behind. Comparing `ER-S` and `ER-S-Tree`, we observe that in the majority of cases `ER-S-Tree` outperforms `ER-S`, which is explained by the superior relaxation strength of `ER-S-Tree` shown in Section 6. The performance improvement increases with higher values of k: While for $k = 3$, both models show almost equal performance, `ER-S-Tree` clearly dominates for $k = \{7, 8, 9, 10\}$. This coincides with the fact that `Tree` performs much better with increasing k. We therefore hypothesize that the performance impact of the subtour elimination constraints increases with the number of regions.

For the generated grid instances (Subfigures 6e - 6h), we observe similar performance trends as for the real-world instances; `ER-S` and `ER-S-Tree` outperform the models `Flow` and `Tree` for all choices of k, while `Flow` outperforms `Tree` for low values of k and `Tree` outperforms `Flow` for high values of k. `ER-S-Tree` outperforms `ER-S` for higher values of k, while for lower values the performance is similar. One notable difference is that for the grid instances, `Tree` has better performance relative to `Flow` when compared to the real-world instances. It seems that `Tree` benefits from a grid graph structure.

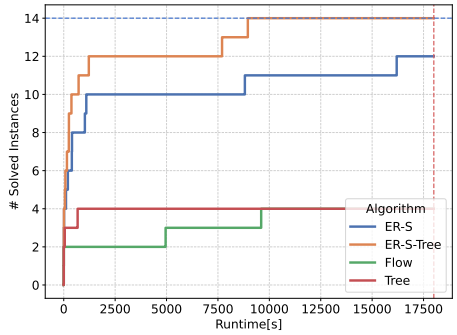
We mention that for instances that were not solved by any model. i.e. grid instances of size 8×8 and 9×9, `ER-S` and `ER-S-Tree` consistently provide much stronger bounds than `Flow` and `Tree`, in many cases achieving optimality gaps of $< 10\%$. The detailed results can be found in the full version of the paper footnoteSee `https://arxiv.org/abs/2607.04886`. For an ablation study of the performance impact of different implementation choices, see Section E of the appendix.



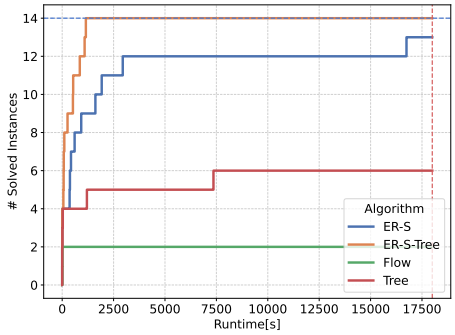
(a) Results for $k \in \{3, 4\}$, real-world instances.



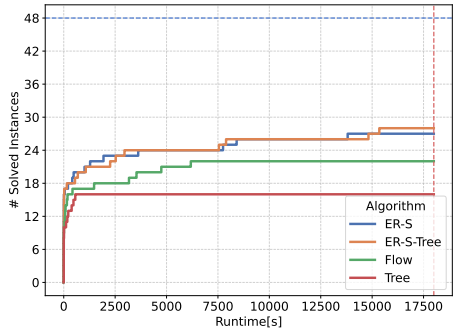
(b) Results for $k \in \{5, 6\}$, real-world instances.



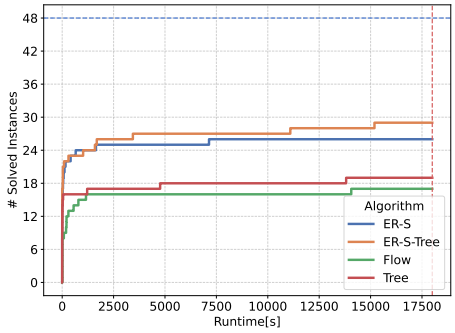
(c) Results for $k \in \{7, 8\}$, real-world instances.



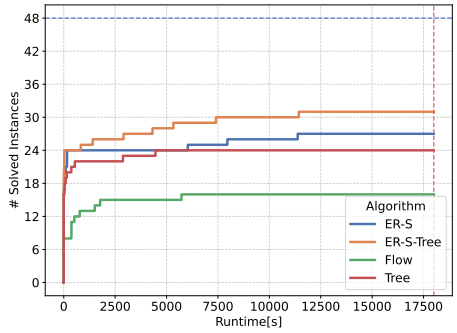
(d) Results for $k \in \{9, 10\}$, real-world instances.



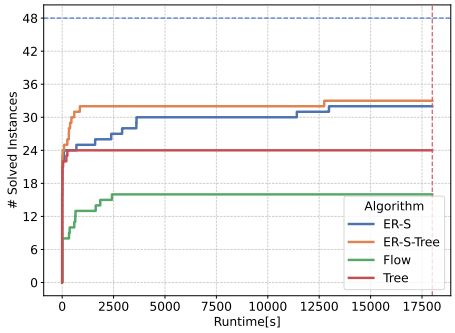
(e) Results for $k \in \{3, 4\}$, grid instances.



(f) Results for $k \in \{5, 6\}$, grid instances.

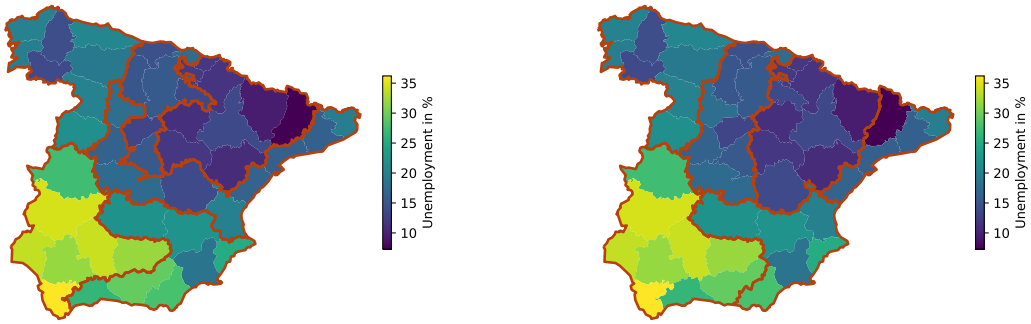


(g) Results for $k \in \{7, 8\}$, grid instances.



(h) Results for $k \in \{9, 10\}$, grid instances.

Figure 6 Performance plot showing the number of solved instances within a given time for `ER-S`, `ER-S-Tree`, `Flow` and `Tree`, grouped by the number of predefined numbers k.



(a) Only considering pairwise feature similarity.

(b) Linear combination between feature similarity and cut edges.

Figure 7 Examples of resulting regionalization on the NUTS-3 subdivison of Spain for $k = 6$ for different costs.

9 Conclusion and Outlook

In this paper, we proposed two new ILP models for the p-regions problem: `ER-S` which models connectivity using vertex separator constraints, and `ER-S-Tree`, which combines vertex-separator constraints and subtour elimination constraints. The model `ER-S-Tree` also includes a new type of subtour elimination constraint, which is specific to the p-regions problem. We conducted a theoretical analysis of the polyhedral strength of each model, proving that `ER-S-Tree` is strictly stronger than the other models. Our computational experiments show that the proposed models `ER-S` and `ER-S-Tree` strongly outperform the models from the literature across all chosen values for k, both on real-world instances and generated grid instances, supporting our theoretical results.

In line with other work [22, 20], we would like to incorporate regional compactness into the problem objective in the future. Notably, the popular compactness criterion of minimizing cut edges [23] can be incorporated without any change to the algorithms; it simply suffices to modify the costs for every vertex pair that is connected by an edge. First experiments showed that including compactness into the objective often decreases the running time of the algorithms, as connected regions emerge more naturally. Figure 7 shows two subdivisions of Spain for different objectives.

For further research, it would be interesting to apply techniques from this paper to the max-p-regions problem [9]. In this variant of the problem, the number of output regions is not fixed, and the granularity of the partition is instead constricted by bounds on the region sizes. As a general direction, our theoretical results show that vertex separator constraints and subtour elimination constraints cut off different fractional solutions from the solution polytope. Combining separator constraints and subtour elimination constraints therefore seems to be a promising direction for a wider range of connectivity constrained problems.

References

1 Tobias Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009. doi:10.1007/s12532-008-0001-1.

2 Zacharie Ales, Arnaud Knippel, and Alexandre Pauchet. Polyhedral combinatorics of the k-partitioning problem with representative variables. *Discrete Applied Mathematics*, 211:1–14, 2016. doi:10.1016/j.dam.2016.04.002.

3 Stefano Benati, Diego Ponce, Justo Puerto, and Antonio M. Rodríguez-Chía. A branch-and-price procedure for clustering data that are graph connected. *Eur. J. Oper. Res.*, 297(3):817–830, 2022. doi:10.1016/J.EJOR.2021.05.043.

4 Stefano Benati, Justo Puerto, and Antonio M. Rodríguez-Chía. Clustering data that are graph connected. *Eur. J. Oper. Res.*, 261(1):43–53, 2017. doi:10.1016/J.EJOR.2017.02.009.

5 Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004. doi:10.1109/TPAMI.2004.60.

6 Sunil Chopra and Mendu R Rao. The partition problem. *Mathematical programming*, 59(1):87–115, 1993. doi:10.1007/BF01581239.

7 Sunil Chopra and M.R. Rao. Facets of the k-partition polytope. *Discrete Applied Mathematics*, 61(1):27–48, 1995. doi:10.1016/0166-218X(93)E0175-X.

8 Juan Duque, Richard Church, and Richard Middleton. The p-regions problem. *Geographical Analysis*, 43:104–126, January 2011. doi:10.1111/j.1538-4632.2010.00810.x.

9 Juan C. Duque, Luc Anselin, and Sergio J. Rey. The max-p-regions problem. *Journal of Regional Science*, 52(3):397–419, 2012. doi:10.1111/j.1467-9787.2011.00743.x.

10 Juan Carlos Duque, Mario C. Vélez-Gallego, and Laura Catalina Echeverri. On the performance of the subtour elimination constraints approach for the p-regions problem: A computational study. *Geographical Analysis*, 50(1):32–52, 2018. doi:10.1111/gean.12132.

11 Statistical office of the European Union. Accessed: 2025-06-03. URL: https://ec.europa.eu/eurostat/web/main/home.

12 Eurostat. NUTS – Nomenclature of territorial units for statistics. https://ec.europa.eu/eurostat/web/nuts. Accessed: 2025-06-03.

13 Daniel Faber. pRegionsERS. https://zenodo.org/records/21275790, 2026. doi:10.5281/zenodo.21275790.

14 Jamie Fairbrother, Adam N. Letchford, and Keith Briggs. A two-level graph partitioning problem arising in mobile wireless communications. *Computational Optimization and Applications*, 69(3):653–676, April 2018. doi:10.1007/s10589-017-9967-9.

15 European Observation Network for Territorial Development and Cohesion. European Regions 2010: Economic Welfare and Unemployment. https://archive.espon.eu/topics-policy/publications/maps-month/european-regions-2010-economic-welfare-and-unemployment. Accessed: 2025-06-03.

16 Patrick Healy, Nicolas Jozefowiez, Pierre Laroche, Franc Marchetti, Sébastien Martin, and Zsuzsanna Róka. A branch-and-cut algorithm for the connected max-k-cut problem. *Eur. J. Oper. Res.*, 312(1):117–124, 2024. doi:10.1016/J.EJOR.2023.06.015.

17 Christopher Hojny, Imke Joormann, Hendrik Lüthen, and Martin Schmidt. Mixed-integer programming techniques for the connected max-k-cut problem. *Math. Program. Comput.*, 13(1):75–132, 2021. doi:10.1007/S12532-020-00186-3.

18 Thorsten Koch and Alexander Martin. Solving Steiner tree problems in graphs to optimality. *Networks*, 32(3):207–232, 1998. doi:10.1002/(SICI)1097-0037(199810)32:3<207::AID-NET5>3.0.CO;2-O.

19 James Lesage. An introduction to spatial econometrics. *Revue d'économie industrielle*, 123, September 2008. doi:10.4000/rei.3887.

20 Wenwen Li, Richard L. Church, and Michael F. Goodchild. An extendable heuristic framework to solve the p-compact-regions problem for urban economic modeling. *Comput. Environ. Urban Syst.*, 43:1–13, 2014. doi:10.1016/J.COMPENVURBSYS.2013.10.002.

21 Thomas L Magnanti and Laurence A Wolsey. Optimal trees. *Handbooks in operations research and management science*, 7:503–615, 1995. doi:10.1016/S0927-0507(05)80126-4.

22 Johannes Oehrlein and Jan-Henrik Haunert. A cutting-plane method for contiguity-constrained spatial aggregation. *J. Spatial Inf. Sci.*, 15(1):89–120, 2017. doi:10.5311/JOSIS.2017.15.379.

23 Hamidreza Validi and Austin Buchanan. Political districting to minimize cut edges. *Math. Program. Comput.*, 14(4):623–672, 2022. doi:10.1007/S12532-022-00221-5.

24 Jay M. Ver Hoef, Erin E. Peterson, Mevin B. Hooten, Ephraim M. Hanks, and Marie-Josèe Fortin. Spatial autoregressive models for statistical inference from ecological data. *Ecological Monographs*, 88(1):36–59, 2018. doi:10.1002/ecm.1283.

A Proofs of polyhedral results

A.1 Proof of Theorem 5

Proof. For $k = n$, simply set $z_{a,a} = 1$ and $r_{a,a}$ and all flow variables to zero. The right-hand side of (1) is always zero and therefore satisfied.

For $3 \leq k \leq n-1$, let $(z, r, x, f) \in P(\texttt{Flow})$ for G, k. For any $i \neq 1$ and $v \neq 1$, set $z_{v,i} = \frac{1}{k-1}$ and set $r_{a,i} = \frac{1}{n-1}$ and all flow variables to zero. This assignment trivially satisfies constraints (2)-(6), (8) and (1) for any value of $x_{u,v}$ (the right-hand side is always < 0).

For constraints (7), the following holds:

$$M \cdot r_{v,i} = \frac{n-k+1}{n-1} = 1 - \frac{k-2}{n-1} \geq 1 - \frac{k-2}{k} = \frac{2}{k} \geq \frac{1}{k-1}$$

where the last inequality follows from $k \geq 3$. Therefore, $z_{v,i} - M \cdot r_{v,i} \leq 0$ and the constraints are always satisfied.

It immediately follows that $P_x(\texttt{ER-S}) \subseteq P_x(\texttt{Flow})$ and $P_x(\texttt{Tree}) \subseteq P_x(\texttt{Flow})$. We now show that these inclusions are strict for $k < n$: Let G be a any graph with $V = \{v_1, \ldots, v_n\}$ and $3 \leq k < n$ and let $z_{a,l} = \frac{1}{k-1}$, $r_{a,l} = \frac{1}{n-1}$ for $a \in V \setminus \{1\}$, $l \geq 2$ and $x_{u,v} = 0$ for $u, v \in V, u < v$ be a solution for `Flow`. This solution is neither contained in `ER-S` nor `Tree`: For `ER-S`, constraint (24) implies $r_v < 1$ for at least one vertex v (as $k < n$). By constraints (23), this implies $\sum_{u<v} x_{u,v} > 0$, contradicting $x_{u,v} = 0$ for all $u < v$. Furthermore, this solution is also not contained in `Tree`, as constraints (16) enforce that $\sum_{e \in A(E)} y_e \leq \sum_{u<v} x_{u,v} = 0$, which contradicts Constraint (13). ◀

A.2 Proof of Theorem 6

Proof. Obviously, it holds that $P_x(\texttt{Tree+}) \subseteq P_x(\texttt{Tree})$ as constraints (29) imply (16) and all other constraints of `Tree` are also part of `Tree+`. To show that the strengthened cycle inequalities (30) can cut off additional fractional solutions, consider the following instance: Let G be a triangle graph with $V = \{v_1, v_2, v_3\}$ and $k = 1$. A valid solution for the fractional relaxation of `Tree` is $x_{1,2} = x_{2,3} = x_{1,3} = y_{1,2} = y_{2,3} = y_{3,1} = \frac{2}{3}$. However, this solution violates constraints (30), as

$$2 \cdot \sum_{(u,v) \in A(E)} y_{u,v} - \sum_{\{u,v\} \in E} x_{u,v} = 4 - 2 \not\leq |V| - 2$$ ◀

A.3 Proof of Observation 4

Proof. Let $S \subseteq V$ and $T \subseteq E(S)$ with $|T| = |S|$ and let (x^*, y^*) be a variable assignment satisfying Constraint (30) for S. Then it holds that:

$$\begin{aligned}
2 \cdot \sum_{(u,v) \in A(T)} y^*_{u,v} &= 2 \cdot \sum_{(u,v) \in A(T)} y^*_{u,v} - |S| + |S| && (34)\\
&\leq 2 \cdot \sum_{(u,v) \in A(T)} y^*_{u,v} - \sum_{\{u,v\} \in T} x^*_{u,v} + |S| && (35)\\
&\leq 2 \cdot (|S| - 1) && (36)
\end{aligned}$$

◀

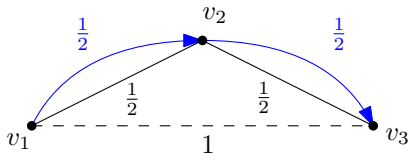


Figure 8 Example for $P(\texttt{Tree}) \not\subset P(\texttt{ER-S})$: $k = 2$, the values of y are in blue, the values of x in black. Solid lines depict existing edges.

$$k = 4$$
$$r_1, r_5 = 1, r_2 = \tfrac{4}{5}, r_3 = \tfrac{3}{5}, r_4 = \tfrac{2}{5}, r_6 = \tfrac{1}{5}$$

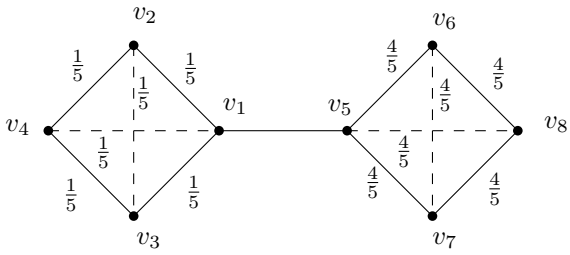


Figure 9 Example for $P(\texttt{ER-S}) \not\subset P(\texttt{Tree})$, $k = 4$: the figure depicts the non-zero values of x, existing edges are drawn as solid lines.

A.4 Proof of Theorem 7

The example consists of the graph with $V = \{v_1, v_2, v_3\}$ $E = \{\{v_1, v_2\}, \{v_2, v_3\}\}$. Consider the following solution $(x, y) \in P(\texttt{Tree})$: Let $k = 2$, $x_{v_1,v_2} = x_{v_2,v_3} = \frac{1}{2}$, $x_{v_1,v_3} = 1$ and $y_{v_1,v_2} = y_{v_2,v_3} = \frac{1}{2}$. The feasibility of this solution is trivial, however there exists no feasible solution $(x, r) \in P(\texttt{ER-S})$. The reason for this is the violation of the separator constraint $x_{v_1,v_3} \not\leq x_{v_1,v_2}$.

To show that $P_x(\texttt{ER-S}) \not\subset P_x(\texttt{Tree})$, we can use the example depicted in Figure 9 for $k = 4$, which is in $P(\texttt{ER-S})$ but not $P(\texttt{Tree})$. First we show that the solution is feasible for ER-S: Constraints (21) and (28) are satisfied as the vertex sets $\{v_1, v_2, v_3, v_4\}$ and $\{v_5, v_6, v_7, v_8\}$ all have the same pairing values within each group and all pairings between the two groups are zero. Furthermore, the values of the representative variables satisfy (22)-(23) and sum up to exactly 4. The pairing variables of adjacent edges sum up to 4, so (31) is also satisfied. Finally, we consider the general clique constraints (32): For $|Q| = 5$, the vertex set with the lowest clique sum consists of $Q_5 = \{v_1, v_2, v_3, v_4, v_5\}$, which sums up to $\frac{6}{5} \geq 1$. For $|Q| = 6$, the lowest sum is obtained with Q_6 being the union of Q_5 and one vertex from v_6, v_7, v_8, which sums up to 2. For Q_8, the total sum of all pairing variables is $6 \geq 4$ and for Q_7, it must be at least $6 - \frac{12}{5} = \frac{18}{5} \geq 3$. To see that there cannot be a valid solution for Tree with such an assignment of pairing variables, observe that $\sum_{\{u,v\} \in E} x_{u,v} = 4 = n - k$, so for $\sum_{(u,v) \in A(E)} y_{u,v} = n - k$ to hold, it must hold that $y_{u,v} + y_{v,u} = x_{u,v}$ for all $\{u, v\} \in E$. However, for the cycle $C = \{v_5, v_6, v_7, v_8\}$, this would imply that $\sum_{(u,v) \in A(E)(C)} y_{u,v} = \frac{16}{5} > 3$, which would violate the cycle inequality.

$P_x(\texttt{Tree}) \not\subset P_x(\texttt{ER-S})$ and $P_x(\texttt{ER-S}) \not\subset P_x(\texttt{Tree+})$ follow from $P_x(\texttt{Tree+}) \not\subset P_x(\texttt{ER-S})$ and $P_x(\texttt{ER-S}) \not\subset P_x(\texttt{Tree})$ and Theorem 6.

B Separation of General Clique Constraints

In this section, we give the pseudocode of the heuristic we use to separate constraints (32). We denote $\bar{x}(u,v)$ to be the fractional value of $x_{u,v}$ in the current solution and define $\bar{x}(u,S) = \sum_{v \in S} x_{u,v}$ for $S \subseteq V \setminus \{u\}$.

Algorithm 1 Seperating General Clique Constraints.

Require: Fractional assignments $\bar{x} : \binom{V}{2} \to \mathbb{R}_{\geq 0}$, Clique size limit $limit$
  Find $u, v \in V$ minimizing $\bar{x}(u,v)$
  $S \leftarrow \{u, v\}$
  while $|S| \leq limit$ **do**
    Let $w \in V \setminus S$ be the vertex minimizing $\bar{x}(w,S)$
    $S \leftarrow S \cup \{w\}$ ▷ Greedily extending S with the best vertex
    Find vertex pairs $a \in V \setminus S$, $b \in S$ minimizing $\Delta(a,b) = \bar{x}(a,S) - \bar{x}(b,S) - \bar{x}(a,b)$
    if $\Delta(a,b) < 0$ **then**
      $S \leftarrow (S \setminus \{b\}) \cup \{a\}$ ▷ Swap b for a
    end if
    Let $|S| = qk + p$
    if $|S| \geq k+1$ **and** $\bar{x}(E(S)) < \binom{q+1}{2}p + \binom{q}{2}(k-p)$ **then**
      Add inequality $x(E(S)) \geq \binom{q+1}{2}p + \binom{q}{2}(k-p)$
    end if
  end while

C Implementation Details

In this section, we give further details on our implementation.

Separation routines

We used the implementation of the Boykov-Kolmogorov flow algorithm [5] provided in `BGL` as a subroutine for computing the minimum cut in the separation of constraints (28) and (16).

For fractional solutions, we only add cuts if they are violated by at least a tolerance ϵ, i.e. a cut $a^t x \leq b$ is added if $a^t x > (1+\epsilon)b$ or in the case of $b = 0$ if $a^t x > \epsilon$. We set $\epsilon = 5\%$

For the general clique inequalities, we set $limit = 3 \cdot k + 3$. An exception is the general clique inequality corresponding to $S = V$, which we add beforehand. We observed that this helps finding a strong initial root relaxation.

To add cutting planes during the solving process, we use the callback function of Gurobi. We add violated inequalities using `addLazy` in integer separation routines, and `addCut` in fractional separation routines. The difference is that constraints added using `addLazy` are enforced in each branch-and-bound node, while constraints added using `addCut` are stored in the global cut pool and are applied selectively by the solver.

Despite the internal cut filters of the solver, we still observed slightly better performance if we prefiltered our cuts before handing them to the solver. For this reason, during the fractional separation routines, we only add the 30% of identified violated cuts that have the highest efficacy [1] (the distance of the hyperplane from the separated invalid point). In the case of integer solutions, all violated separator constraints are added.

Reimplementation of algorithms

The authors in [8] did not provide the source code for their algorithms, thus we reimplemented the algorithms following the descriptions in [8] and the follow-up paper [10] as closely as possible. For `Tree`, we used `addLazy` to separate violated cycle-breaking inequalities in each integer callback. Following [10] we used the `BGL` implementation of Tarjan's algorithm to calculate strongly connected components. For each strongly connected component Γ containing more than one vertex, we add the corresponding inequality of type (16).

We used default parameters for `Gurobi` (with the exception of setting `PreCrush=1`, as this is necessary to guarantee user cuts working correctly).

D Generation of random instances

For the generation of the grid instances, we generated the vertex attributes simulating a spatial auto-regressive model (see for example [24]): Given an integer $Size$ and an autocorrelation parameter ρ, the generated graph is a grid graph of size $Size \times Size$. The vertex attributes are given as $\Gamma_{G,\rho} = ((I - \rho\bar{A})^{-1})^t\epsilon$, where I is the identity matrix, $\bar{A}$ is the row-normalized adjacency matrix of G, and ϵ is a vector of entries drawn from a normal distribution. The vertex dissimilarities between vertices u and v are now defined as $d(u,v) = |(\Gamma_{G,\rho})_u - (\Gamma_{G,\rho})_v|$, $(\Gamma_{G,\rho})_u$ representing the matrix entry corresponding to vertex u.

E Ablation study

We evaluated the performance impact of our implementation decisions on the limited benchmark set, additionally including the instance `Romania`. For each of the tested feature, we ran the model with this specific feature deactiviated. The tested features where the following:

- (Leaving out) Filter: Filtering out the cuts with the highest efficacy was deactivated, i.e. all identified cuts where added
- (Leaving out) Local Search (LocSrch): The local search step in the separation of the general clique inequalities was skipped, i.e. the separation algorithm exclusively performs the greedy extension step in each iteration
- (Leaving out) Nested Cuts (Nested): We do not perform nested cuts, only adding at most one cut to each vertex pair in each iteration
- (Leaving out) strengthening inequalities (SI): We omit adding (31) and the general clique inequality for $S = V$ to the initial model

The results are shown in Figure 10. `ER-S-Full` denotes `ER-S`, i.e. all features enabled. We can see that for the relatively easy instances, the performance is close to equal, while for the harder instances, we see some notable differences in performance. Disabling the local search step in the general clique separation seems to yield the largest performance degradation, followed by disabling strengthening inequalities and cut filtering. On the other hand, the performance benefit of using nested cuts is not completely clear, as there are multiple instances that are solved faster when nested cuts are disabled. Nevertheless, we still chose the current setting as the full model solves the most instances within a 5 hour time limit.

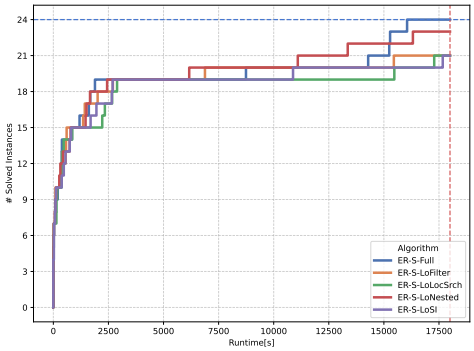


Figure 10 Leave-one-out experiment for different implemented features.

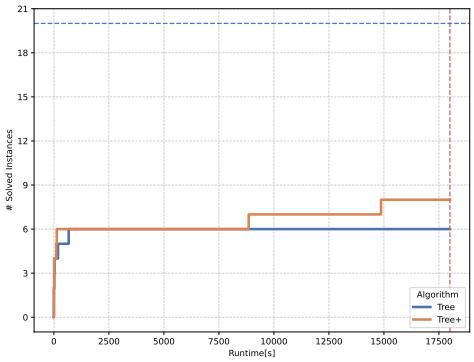


Figure 11 Performance comparison of `Tree` and `Tree+`.

F Experimental comparison of `Tree` and `Tree+`

We conducted a computational comparison between `Tree` and `Tree+` on the limited benchmark set. The results can be seen in Figure 11. We can see that `Tree+` outperforms `Tree`, solving two more instances.